Вестник

Российско-Армянского (Славянского) университета

№2

ФИЗИКО-МАТЕМАТИЧЕСКИЕ И ЕСТЕСТВЕННЫЕ НАУКИ

ISSN 1829-0450

ЕРЕВАН 2015

РОССИЙСКО-АРМЯНСКИЙ (СЛАВЯНСКИЙ) УНИВЕРСИТЕТ

В Е С Т Н И К

РОССИЙСКО-АРМЯНСКОГО (СЛАВЯНСКОГО) УНИВЕРСИТЕТА

СЕРИЯ:

ФИЗИКО-МАТЕМАТИЧЕСКИЕ И ЕСТЕСТВЕННЫЕ НАУКИ

№2

Издательство РАУ

Ереван 2015

ՀԱՅ-ՌՈՒՍԱԿԱՆ (ՍԼԱՎՈՆԱԿԱՆ)
ՀԱՄԱԼՍԱՐԱՆ

Լ Ր Ա Բ Ե Ր

ՀԱՅ-ՌՈՒՍԱԿԱՆ (ՍԼԱՎՈՆԱԿԱՆ)

ՀԱՄԱԼՍԱՐԱՆԻ

ՍԵՐԻԱ
ՖԻԶԻԿԱՄԱԹԵՄԱՏԻԿԱԿԱՆ

ԵՎ ԲՆԱԿԱՆ ԳԻՏՈՒԹՅՈՒՆՆԵՐ

№ 2

ՀՌՀ Հրատարակչություն

Երևան 2015

Печатается по решению Ученого совета РАУ

Вестник РАУ, № 2. – Ер.: Изд-во РАУ, 2015. – 107 с.

Журнал входит в перечень периодических изданий, зарегистрированных ВАК РА

Российско-Армянский (Славянский) университет, 2015 г.

ISBN 1829-0450

МАТЕМАТИКА И ИНФОРМАТИКА

УДК 539.3 Поступила 11.09.2015г.

К ЗАДАЧЕ ПЛАНАРНЫХ КОЛЕБАНИЙ ПЛАСТИНКИ

С.А. Амбарцумян, М.В. Белубекян

Институт механики НАН РА
e-mail: mechins@sci.am

АННОТАЦИЯ

Предлагается вариант уточнения задач планарных колебаний упругой пластинки. Поперечные напряжения представлены в виде квадратичной функции по толщине пластинки. Исследуются эффекты уточнения в завимости от условий закрепления кромок пластинки.

Ключевые слова: упругая пластинка, продольная волна, закрепленный край.

Исследование пространственных (трехмерных) задач пластин связано с большими трудностями [1]. Наиболее известны четыре метода сведения трехмерных задач пластин (и оболочек) к двумерным: а) метод определения или метод гипотез [2], б) асимптотический метод [3, 4], в) вариационный метод [5, 6], г) операторный метод [7].

Основным недостатком метода гипотез принято считать невозможность продолжения процесса уточнения. Здесь, на частном примере, показывается, что процесс уточнения (процесс получения следующих приближений) возможен. Однако, при этом, необходимо одновременно использовать как процедуру увеличения количества членов разложения [8] по толщинной координате, так и процедуру осереднения уравнений движения с использованием моментов более высоких порядков.

В настоящей статье учет нормальных по толщине напряжений уточняет задачу планарных колебаний, но не влияет на изгибные колебания.

1. **Рассматривается задача колебаний прямоугольной пластинки**, которая в прямоугольной декартовой системе координат занимает область: $0 \le x \le a,\ 0 \le y \le b,\ -h \le z \le h$.

Пространственные уравнения движения пластинки имеют вид:

$$\frac{\partial \sigma_{ij}}{\partial x_j} = \rho \frac{\partial^2 u_i}{\partial t^2,}, \; i = 1, 2, 3 \tag{1.1}$$

Лицевые поверхности пластинки свободны

$$\sigma_{3i} = 0 \;\; при \;\; z = \pm h \tag{1.2}$$

В законе Гука, в отличие от теории Кирхгофа, напряжение σ_{33} не пренебрегается, а принимается в виде:

$$\sigma_{33} = \left(1 - \frac{z^2}{h^2}\right)\varphi_3(x, y, t) = f(z)\varphi_3 \tag{1.3}$$

Напряжения σ_{31}, σ_{32} в законе Гука пренебрегаются для простоты. Можно для них также использовать некоторые представления [2]. Здесь относительно $\sigma_{31,}\sigma_{32}$ останавливаемся на уровне теории Кирхгофа, так как их уточнения первого и высоких порядков, как известно, [2, 9] не будет влиять на планарные колебания.

С помощью закона Гука и допущения (1.3) для основных напряжений имеем:

$$\begin{gathered}
\sigma_{11} = \frac{E}{1-\nu^2}(\varepsilon_{11} + \nu\varepsilon_{22}) + \frac{\nu}{1-\nu} f(z)\varphi_3 \\
\sigma_{22} = \frac{E}{1-\nu^2}(\varepsilon_{22} + \nu\varepsilon_{14}) + \frac{\nu}{1-\nu} f(z)\varphi_3 \\
\sigma_{12} = \frac{E}{1+\nu}\varepsilon_{12}, \;\; \varepsilon_{ij} = \frac{1}{2}\left(\frac{\partial u_i}{\partial x_j} + \frac{\partial j}{\partial x_i}\right).
\end{gathered} \tag{1.4}$$

Для перемещений, как и в теории Кирхгофа, принимается

$$u_1 = u - z\frac{\partial w}{\partial x}, \; u_2 = \mathrm{v} - z\frac{\partial w}{\partial y}, \; u_3 = w, \tag{1.5}$$

где функции u, v, w не зависят от координаты z.

В классической теории уравнения (1.1) осередняются интегрированием по толщине пластинки для $i = 1, 2, 3$ и интегрированием по толщине после

умножения на z для $i=1,2$ (моменты первого порядка). Здесь, в дополнение к классической теории, процесс осреднения продолжается. Уравнение (1.1) при $i=3$ также осредняется умножением на z, затем уравнения (1.1) при $i=1,2$ умножаются на z^2 и снова осредняются (моменты второго порядка). В результате получаются следующие уравнения в усилиях и моментах:

$$\frac{\partial T_1}{\partial x}+\frac{\partial S}{\partial y}=2\rho h\frac{\partial^2 u}{\partial t^2}\,,\quad \frac{\partial T_2}{\partial y}+\frac{\partial S}{\partial x}=2\rho h\frac{\partial^2 \mathrm{v}}{\partial t^2} \tag{1.6}$$

$$\begin{gathered}\frac{\partial N_1}{\partial x}+\frac{\partial N_2}{\partial y}=2\rho h\frac{\partial^2 w}{\partial t^2}\\ \frac{\partial M_1}{\partial x}+\frac{\partial H}{\partial y}-N_1=0,\ \frac{\partial M_2}{\partial y}+\frac{\partial H}{\partial x^2}-N_2=0\end{gathered} \tag{1.7}$$

$$\frac{\partial N_{11}}{\partial x}+\frac{\partial N_{21}}{\partial y}=\frac{4h}{3}\varphi_3\,,$$

$$\frac{\partial T_{12}}{\partial x}+\frac{\partial S_2}{\partial y}-2N_{11}=\frac{2\rho h^3}{3}\frac{\partial^2 u}{\partial t^2},\quad \frac{\partial T_{12}}{\partial y}+\frac{\partial S_2}{\partial x}-2N_{21}=\frac{2\rho h^3}{3}\frac{\partial^2 \mathrm{v}}{\partial t^2}. \tag{1.8}$$

В отличие от классической теории, уравнения (1.7) определяют изгибные колебания, которые здесь не рассматриваются. Усилия и моменты второго порядка в уравнениях (1.6) и (1.8) определяются с учетом (1.2), (1.4), (1.5) следующим образом:

Второе и третье уравнения (1.8) напоминают дополнительные уравнения движения микрополярной теории упругости.

Уравнения (1.7) определяют изгибные колебания, которые здесь не рассматриваются. Усилия и моменты второго порядка в уравнениях (1.6) и (1.8), в отличие от классической теории, определяются с учетом (1.2), (1.4), (1.5) следующим образом:

$$T_1=C\left(\frac{\partial u}{\partial x}+\nu\frac{\partial \mathrm{v}}{\partial y}\right)+\frac{4\nu h}{3(1-\nu)}\varphi_3\,,\ S=\frac{1-\nu}{2}C\left(\frac{\partial u}{\partial y}+\frac{\partial \mathrm{v}}{\partial x}\right);$$

$$T_2=C\left(\frac{\partial \mathrm{v}}{\partial y}+\nu\frac{\partial u}{\partial x}\right)+\frac{4\nu h}{3(1-\nu)}\varphi_3\,,\ C=\frac{2Eh}{1-\nu^2} \tag{1.9}$$

$$T_{12}=\int_{-h}^{h} z^2\sigma_{11}dz = D\left(\frac{\partial u}{\partial x}+\mathrm{v}\frac{\partial \mathrm{v}}{\partial y}\right)+\frac{4vh^3}{15(1-v)}\varphi_3$$

$$T_{22}=\int_{-h}^{h} z^2\sigma_{22}dz = D\left(\frac{\partial \mathrm{v}}{\partial y}+v\frac{\partial u}{\partial x}\right)+\frac{4vh^3}{15(1-\mathrm{v})}\varphi_3 \qquad (1.10)$$

$$S_2=\frac{1-\mathrm{v}}{2}D\left(\frac{\partial u}{\partial y}+v\frac{\partial \mathrm{v}}{\partial x}\right),\ D=\frac{2Eh^3}{3\left(1-\mathrm{v}^2\right)}$$

С учетом (1.10) моменты N_{11} и N_{21} определяются из второго и третьего уравнения системы (1.8).

$$N_{11}=\frac{Eh^3}{6(1+v)}\left[\Delta u+\theta\frac{\partial}{\partial x}\left(\frac{\partial u}{\partial x}+\frac{\partial v}{\partial y}\right)+\frac{4v\theta}{5E}\frac{\partial\varphi_3}{\partial x}-\frac{1}{C_t^2}\frac{\partial^2 u}{\partial t^2}\right]$$

$$N_{21}=\frac{Eh^3}{6(1+v)}\left[\Delta \mathrm{v}+\theta\frac{\partial}{\partial y}\left(\frac{\partial u}{\partial x}+\frac{\partial \mathrm{v}}{\partial y}\right)+\frac{4v\theta}{5E}\frac{\partial\varphi_3}{\partial y}-\frac{1}{C_t^2}\frac{\partial^2 \mathrm{v}}{\partial t^2}\right] \qquad (1.11)$$

$$C_t^2=\frac{E}{2\rho(1+v)},\ \theta=\frac{1+v}{1-v}$$

Подстановка (1.10) в систему (1.6) и (1.11) в первое уравнение системы (1.8) приводит к системе уравнений относительно трех искомых функций u, v, φ_3:

$$\Delta u+\theta\frac{\partial}{\partial x}\left(\frac{\partial u}{\partial x}+\frac{\partial \mathrm{v}}{\partial y}\right)+\frac{4v\theta}{3E}\frac{\partial\varphi_3}{\partial x}=\frac{1}{C_t^2}\frac{\partial^2 u}{\partial t^2}$$

$$\Delta \mathrm{v}+\theta\frac{\partial}{\partial y}\left(\frac{\partial u}{\partial x}+\frac{\partial v}{\partial y}\right)+\frac{4v\theta}{3E}\frac{\partial\varphi_3}{\partial y}=\frac{1}{C_t^2}\frac{\partial^2 \mathrm{v}}{\partial t^2}$$

Подстановка (1.10) в систему (1.6) и (1.11) в первое уравнение системы (1.8) приводит к системе уравнений относительно трех искомых функций u, v, φ_3

$$\Delta\left(\frac{\partial u}{\partial x}+\frac{\partial \mathrm{v}}{\partial y}\right)-\frac{\rho\left(1-v^2\right)}{E}\frac{\partial^2}{\partial t^2}\left(\frac{\partial u}{\partial x}+\frac{\partial \mathrm{v}}{\partial y}\right)+\frac{2v(1+v)}{5E}\Delta\varphi_3-\frac{4(1-v^2)}{Eh^2}\varphi_3=0 \quad (1.12)$$

2. **Граничные условия**. Согласно (1.12), в отличие от теории Кирхгофа, на кромках пластины $x=const$, $y=const$ для задачи планарных колебаний

(обобщенного плоского напряженного состояния необходимо иметь три граничных условия). В случае закрепленной кромки пластинки добавочное условие, естественно, нужно взять в виде:

$$\int_{-h}^{h} \varepsilon_{33} dz = 0\ ,\ \ \varepsilon_{33} = \frac{1}{E}\left[\sigma_{33} - \nu\left(\sigma_{11} + \sigma_{22}\right)\right]. \tag{2.1}$$

Условие (2.1), с учетом (1.4), представляется следующим образом

$$\frac{4h}{3}\varphi_3 - \nu\left(T_1 + T_2\right) = 0. \tag{2.2}$$

Используя выражения для T_1, T_2 из (1.9), условия закрепленного края при $x = const$ приводятся к виду:

$$u = 0.\ \mathrm{v} = 0.\ \nu E\frac{\partial u}{\partial x} - \frac{2}{3}\left(1+\nu\right)\left(1-2\nu\right)\varphi_3 = 0. \tag{2.3}$$

Условия, соответствующие условиям свободного опирания (условиям Навье в трехмерной задаче), очевидно, будут:

$$T_1 = 0,\ \mathrm{v} = 0,\ T_{12} = 0\ \ \textit{при}\ \ x = const. \tag{2.4}$$

Не трудно показать, что условия (2.4) приводятся к виду:

$$\frac{\partial u}{\partial x} = 0,\ \mathrm{v} = 0,\ \varphi_3 = 0. \tag{2.5}$$

Аналогично, при $x = const$ условия скользящего контакта примут вид

$$u = 0,\ S = 0,\ \int_{-h}^{h} \varepsilon_{33} dz = 0 \tag{2.6}$$

или

$$u = 0,\ \frac{\partial v}{\partial x} = 0,\ \nu E\left(\frac{\partial u}{\partial x} + \frac{\partial \mathrm{v}}{\partial y}\right) - \frac{2}{3}\left(1+\nu\right)\left(1-2\nu\right)\varphi_3 = 0 \tag{2.7}$$

Условия свободного края:

$$T_1 = 0,\ S = 0,\ T_{12} = 0 \tag{2.8}$$

или

$$\frac{\partial u}{\partial x}+\nu\frac{\partial \mathrm{v}}{\partial y}=0,\ \frac{\partial u}{\partial y}+\frac{\partial \mathrm{v}}{\partial x}=0,\ \varphi_3=0 \tag{2.9}$$

В частном случае одномерных колебаний система уравнений (1.12), после некоторых преобразований, приводится к виду $\left(\partial/\partial y=0\right)$

$$\frac{\partial^2 u}{\partial x^2}+\beta\frac{\partial \varphi_3}{\partial x}=\frac{1}{C_2^2}\frac{\partial^2 u}{\partial t^2},\ \frac{\partial^2 \varphi_3}{\partial x^2}+\frac{\gamma^2}{h^2}\varphi_3=0,\ \frac{\partial^2 \mathrm{v}}{\partial x^2}=\frac{1}{C_t^2}\frac{\partial^2 \mathrm{v}}{\partial t^2}$$
$$\beta=\frac{2\nu\left(1+\nu\right)}{3E},\ \gamma^2=\frac{15\left(1-\nu\right)}{\nu} \tag{2.10}$$

Уравнения для поперечного перемещения v отделяются. В вышеприведенных вариантах граничных условий перемещение v также отделяется. Уравнение (2.10) также показывает, что, если на кромках пластинки $x=0,a$ заданы либо условия свободного опирания (2.5), либо свободного края (2.9), следует $\varphi_3\equiv 0$. Т.е. для таких задач предлагаемое уточнение не имеет влияния.

Пусть, на краях заданы следующие условия (консольная пластинка)

$$u=0,\ \nu E\frac{\partial u}{\partial x}-\frac{2}{3}\left(1+\nu\right)\left(1-2\nu\right)\varphi_3=0\quad при\ x=0$$
$$\frac{\partial u}{\partial x}=0,\ \varphi_3=0\qquad при\ x=a \tag{2.11}$$

Здесь при $x=0$ есть условия закрепленного края (2.3), а при $x=a$ – свободного края (2.9).

Общее решение гармонических колебаний уравнений (2.10) относительно функций u,v будет:

$$u=\left[A_1\sin\frac{\omega}{C_2}x+B_1\cos\frac{\omega}{C_2}x-\frac{\beta\gamma}{h}\left(\frac{\omega^2}{C_2^2}-\frac{\gamma^2}{h^2}\right)^{-1}\left(A_2\cos\frac{\gamma}{h}x-B_2\sin\frac{\gamma}{h}x\right)\right]e^{iwt}$$
$$\varphi_3=\left(A_2\sin\frac{\gamma}{h}x+B_2\cos\frac{\gamma}{h}x\right)e^{iwt}, \tag{2.12}$$

где $A_1.B_1,A_2,B_2$ – произвольные постоянные.

Подстановка (2.12) в граничные условия (2.11) приводит к системе однородных алгебраических уравнений относительно произвольных постоян-

ных. Условие равенства нулю детерминанта этой системы дает следующее дисперсионное уравнение:

$$\Omega tg\Omega = \zeta tg\zeta, \qquad (2.13)$$

где

$$\Omega = \frac{\omega a}{C_2}, \quad \zeta = \frac{\zeta a}{h}. \qquad (2.14)$$

В классической теории одномерные планарные колебания пластинки приводят к результату:

$$\cos\Omega = 0 \quad \textit{или} \quad \Omega = \frac{(2n-1)}{2} \qquad (2.15)$$

Очевидно, частоты, определяемые из (2.13), будут существенно отличаться от (2.15). В частности, (2.13) имеет также следующее решение:

$$\Omega = \zeta \ \textit{или} \ \frac{\omega}{C_2} = \frac{\zeta}{h} \qquad (2.16)$$

3. **Разделение волн**. Используя преобразование Ламе для плоской задачи [11]

$$u = \frac{\partial\varphi}{\partial x} + \frac{\partial\psi}{\partial y}, \quad v = \frac{\partial\varphi}{\partial y} - \frac{\partial\psi}{\partial x} \qquad (3.1)$$

систему уравнений (1.12) можно привести к виду:

$$\Delta\varphi + \beta\varphi_3 = \frac{1}{C_2^2}\frac{\partial^2\varphi}{\partial t^2}, \ \Delta\Psi = \frac{1}{C_t^2}\frac{\partial^2\psi}{\partial t^2}$$
$$\Delta^2\varphi - \frac{1}{C_2^2}\frac{\partial^2}{\partial t^2}\Delta\varphi + \frac{3(1-\nu)}{10}\alpha\Delta\varphi_3 - \frac{4(1-\nu^2)}{Eh^2}\varphi_3 = 0 \qquad (3.2)$$

Отсюда следует, что уравнение для планарных сдвиговых волн отделяется. Подстановка оператора для φ из первого уравнения (3.1) в третье приводит к раздельному уравнению для φ_3

$$\Delta\varphi_3 + \frac{\gamma^2}{h^2}\varphi_3 = 0 \qquad (3.3)$$

Из (3.1) видно, что, как и в одномерной задаче, если на всех кромках пластины имеется условие $\varphi_3 = 0$, результаты решения задач будут совпадать с классическими.

ЛИТЕРАТУРА

1. *Пикуль В.В.* Механика оболочек. Владивосток. Изд-во ДВГТУ, 2005. 524 с.
2. *Амбарцумян С.А*. Теоррия анизотропных пластин. М.: Наука, 1987. 360 с.
3. *Гольденвейзер А.Л.* Теория упругих тонких оболочек. М.: Наука, 1976. 512 с.
4. *Aghalovyan L.A.* Asymptotic Theory of Anisotropic Plates and Shells. World Scientific. 2015. 360 p.
5. *Бердичевский В.Л.* Вариационные методы построения моделей оболочек.- «Прикл. мат. и мех». 1972, т. 36, вып 5. СС. 788–804.
6. *Векуа И.Н*. О двух путях построения непротиворечивой теории упругих оболочек // В кн.: Материалы I-ой Всес. школы по теории и численным методам расчета оболочек м пластин. Тбилиси, 1975. СС. 5–50.
7. *Лурье А.И*. Пространственные задачи теории упругости. М. Гостехиздат, 1955. 491 с.
8. *Киракосян Р.М*. Прикладная теория ортотропных пластин переменной толщины, учитывающая влияние деформаций поперечных сдвигов. Ер., Изд-во «Гитутюн» НАН РА, 2000. 122 с.
9. *Reddy J.N*. Mechanics of Laminated Composite Plates and Shells. Theory and analiyses.-2nd ed. – Boca Ration.CRC Press. LLC. 2004. 831 p.
10. *Амбарцумян С.А., Белубекян М.В*. Прикладная микрополярная теория упругих оболочек. Ер., Изд-во «Гитутюн» НАН РА, 2010. 136 с.

ON THE PROBLEM OF A PLATE PLANAR VIBRATION

S. Ambartsumian, M. Belubekyan

SUMMARY

The variant of the specifying (improving) of the tlastic plate planar vibration problem is suggested. The transversal stresses are prezent in the form of squpre function along plate thickness. The effects of the specifying are investigated in dependence from the fasten conditions of the plate dges.
Keywords: elastic plate, longitudinal wave, clumped edge.

ՍԱԼԻ ՊԼԱՆԱՐ ՏԱՏԱՆՈՒՄՆԵՐԻ ԽՆԴԻՐԸ

Ս.Ա. Համբարձումյան, Մ.Վ. Բելուբեկյան

ԱՄՓՈՓՈՒՄ

Առաջարկված է առաձգական սալերի պլանար տատանումների խնդիրների ճշգրտման տարբերակ: Լայնական լարումները ներկայացվում են ըստ սալի հաստության քառակուսային ֆունկցիայի տեսքով: Հետազոտվում է ճշգրտման արդյունավետությունը կախված սալի կողերի ամրացման պայմաններից:
Հիմնաբառեր՝ առաձգական սալ, երկայնական ալիք, ամրակցված եզր:

УДК 004.056.55 Поступила 24. 09.2015г.

A GENERIC FRAMEWORK FOR SECURE COMPUTATIONS

D. Danoyan[1], T. Sokhakyan[2]

[1]*Yerevan State University,Yerevan, Armenia, danoyan@gmail.com*

[2]*Russian-Armenian (Slavonic) University, Yerevan, Armenia, tigran.sokhakyan@gmail.com*

SUMMARY

In this article we present a general purpose secure two party computation framework offering security against semi-honest threat model. The framework includes its own language for higher level function description and a compiler translating it into a Boolean circuit. Our compiler can generate large circuits using smaller computational resources than similar compilers and it utilizes the presence of multiple CPUs. We use white-box cryptography based oblivious transfer protocol to avoid expensive public key operations.

Keywords: Secure function evaluation, secure computation protocols, Yao's garbled circuits protocol, white-box cryptography, oblivious transfer.

1. Introduction

As the world is getting more and more connected in many real world scenarios, parties with different and potentially conflicting interests have to interact. Examples of this are citizens and governments (electronic passport and electronic id), patients and health insurers or medical institutions (electronic health card, e-health services), or companies and service providers (cloud computing). In the mentioned context questions of paramount importance are the architecture and the ability of an underlying communication system to fulfill varied security and privacyrequirements of the involved parties.

Protocols for secure computations allow two or more mutually distrustful parties to communicate and compute some commonly agreed function value on each other's input, with privacy and authenticity guarantees. Andrew Yao pointed out that secure two-party protocols can be constructed for computation of any computable function [1].

Yao’s protocol remains one of the most actively studied methods for secure computations. Although Yao never published a precise protocol, the very first real world implementation of secure two-party computation (2PC) [2] used Yao’s basic garbled circuit approach, and it remains the primary paradigm for plenty of 2PC implementations that have been developed during the past eleven years [3][4][5].

Yao’s protocol has a great practical significance. In many real-world situations, the inputs to a function may be too valuable or sensitive to share with other parties. Efficient 2PC algorithms enable a variety of electronic transactions, previously impossible due to mutual mistrust of participants. Huang et al. explored the use of secure computation for biometric identification[6] in security applications, when it is desirable for individual genetic data to be kept private but still checked against a specified list. The more general case of multiparty computation has already seen real-world use in computing market clearing prices in Denmark[7]. This is not so forth the full list of applications: auctions [8], contract signing [9],etc.

2. Background

In this section we briefly introduce the main cryptographic tools we have used in our framework: garbled circuits, white-box cryptography based oblivious transfer (OT) protocol and optimization techniques for Yao’s protocol.

2.1. Yao’s garbled circuits protocol

Yao’s garbled circuit protocol[1] allows two mutually distrustful parties holding inputs x and y to evaluate an arbitrary computable function $f(x, y)$ on their input values without leaking any side information about their inputs beyond what is explicitly implied by the function output.

The main idea is that one party (called garbled circuit *generator*) generates an “encrypted” version of the Boolean circuit Ccomputing the function f, and the second party (called garbled circuit *evaluator*) obliviously computes the garbled circuit. Note that reverse engineering techniques are not applicable to garbled circuit, thus the *evaluator* does not learn any intermediate value.

Suppose the *generator* has a Boolean circuit C with 2 fan-in gates computing the function f. At the first step the *generator* fixes some integer k and assigns two random looking bit strings w^0 and w^1 to each wire of circuit C (label w^b conceptually encodes value $b \in \{0, 1\}$ for the wire w). Then for the gate g having output

wire w_k and input wires w_i, w_j *generator* prepares garbled table with following entries:

$$Enc_{w_i^{b_i}, w_j^{b_j}} \left(w_k^{g(b_j, b_j)}\right), \tag{1}$$

where Enc is an encryption scheme fixed by *generator*. The collection of all garbled gates is called garbled circuit.

Then the *generator* passes garbled circuit and mapping for the labels for the output wires to the *evaluator*. Note that only the *generator* knows mapping between binary input bits for input wires and he can simply send the garbled circuit to the evaluator with label $w_i^{x_i}$ for input wire w_i, where x_i is the i-th bit of his input. To obtain wire labels for his input the *evaluator*runs OT protocol described next with the *generator*.

The evaluation of garbled circuit is done in a hierarchical way. Given labels $w_i$and w_j of input wires of garbled gate g the *evaluator* decrypts the appropriate entry of garbled table using keys $w_i$and w_j. When labels of all output wires are computed the *evaluator* sends the function output value to the *generator* using provided mapping for output wires.

2.2. White-box cryptography based OT protocol

1-out-of-2 oblivious protocol (OT) is an essential part of Yao's garbled circuit protocol. It involves two parties: *sender* holding two strings w_0 and w_1, and *receiver* holding selection bit b. OT protocol allows *sender*to transmit exactly one input string w_b to *receiver*; *receiver* learns nothing about $w_{b\oplus 1}$and *sender* does not learn selection bit b. Currently several OT protocols are available. In our implementation we use novel white-box cryptography based OT protocol which is secure in the semi-honest setting and is introduced in [10].

2.3. Optimization techniques

To get more practical use of this protocol many optimizations have been developed during the past decade, some of which are compatible with each other and are used in our framework. Kolesnikov and Schneider [11] introduced a technique eliminating the need to garble XOR gates (XOR gates become "free", involving no communication or cryptographic operations). Also, we use the technique proposed by Pinkas et al. [12] allowing to reduce the size of a garbled table from four to three

ciphertexts, and saving 25% of network bandwidth for non-XOR gates. Another optimization to apply is the FleXOR technique by Kolesnikov et al. [13] combined with two garbled row reductions [12] instead of one. We have added this option, as two garbled row reductions and FreeXOR techniques are not compatible. The combination of optimization techniques is user configurable.

3. Secure function evaluation framework description

Our implementation follows traditional development paradigms of SFE frameworks described in[2].Like it is done in Fairplay, our framework contains a circuit compiler generating single pass Boolean circuit from the algorithm description given in a higher level language.Popular $flex$ and $bison$ tools were used to generate our compiler.

3.1. Language description

The input language of our compiler is an imperative one with static scoping. The syntax of the language allows writing algorithms with complicated logic without too much extra syntax. Declarations of functions are allowed, but they must be non-recursive for the reason of security.Arrays are also included in our language, but the elements' count and access indices must be constants or determined at compile time. All cycles are unrolled during compilation, thus cycle variable can be used as an array index. Variables can be concatenated; bits or range of bits can be selected or may be assigned to. Also, our compiler supports all types of bitwise operations. We use single assignment algorithm from Fairplay to deal with assignments within if statements effectively.

3.2. Compiler implementation description

The goal of the compiler is to generate maximum possible amount of XOR gates to gain full advantage of Free XOR technique. All circuit blocks generated by the compiler are locally optimized to meet this requirement. To obtain the best result we have used Jegalkin polynomials [14] intensively. Also, our compiler can generate special types of circuits depending on the sizes of operands. For example, when translating multiplication of two 512 bit integers Karatsuba-Ofman multiplication [15] is more likely to be used instead of the standard multiplication circuit to produce more efficient output.

Fairplay compiler requires a huge amount of computational resources to generate circuits having far less size than real world applications. The main goal of our project is to create a general purpose framework for secure computations which is open to modifications and further experiments. Thus, writing an application specific framework for circuit generation, the technique which is used in[16], is not an acceptable option for us. To resolve resource consumption problems, we have implemented a brand new compiler generating a more efficient output than Fairplay does. For example, Fairplay encourages the programmer to count the number of 1s in a 520 bit array by a simple counting algorithm which leads to a circuit that uses 10 bit accumulator during all computations, but a narrower accumulator is sufficient for early stages; in our implementation the accumulator is getting wider as it is needed. This optimization results in a more efficient Hamming distance circuit which is a widely used sub circuit in many applications.

Our compiler requires a fixed amount of memory and utilizes availability of multiple CPUs. To reach these goals we havesplit the file writer in a separate running thread of execution and the gate generator in another one. The connection between them is done via a circular queue. The compiler has configuration options weather to use non-blocking or with mutual blocking queue. Running on computer having at least two CPUs non-blocking queue is more efficient and all processing resources of two CPUs are utilized. As a result, we get multiple running threads and preliminarily fixed memory consumption.

3.3. Compiler optimizations

The gates are emitted by the front-end part of our compiler as the processing of a single expression is completed, thus it contains the number of identity gates i.e. gates whose output value is independent from values of input wires. To address this problem our compiler also includes an optimizer back end. The elimination of thiskind of gates is an important step of the optimization process and has influence in both achieving less network communication (redundant gates are not transferred via network) and doing unnecessary computations during circuit evaluation. Inverter gates are also removed in this step. The removal of gates having no impact on output gates is the last step of optimization of the code generated by the front-end of our compiler. This step is identical to the elimination of the dead code which is an essential step of every industrial compiler. To handle all these gate removal steps efficiently (in terms of both memory consumption and processing time) we generate

and keep external tables (i.e. gate usage count, gate placement position, etc.) and these tables are passed from every step of optimization to the next one. Processing these tables in memory increases memory consumption, and to overcome this problem we make use of memory mapped files intensively. In future we plan to do experiments on various types of storage for these tables: Berkeley DB and custom SQL-like databases.

3.4. Framework description

Our framework used 80 bit wire labels for garbled circuits. SHA-2 is used to generate the garbled truth-table entries as follows:

$$Enc^{k}_{w_i^{b_i}, w_j^{b_j}} \left(w_k^{g(b_i,b_j)}\right) = SHA-2\left(w_i^{b_i} \,||\, w_j^{b_j} \,||k\right) \oplus w_k^{g(b_i,b_j)},$$

where notations are the same as in (1).

Whenever it is possible Intel Advanced Encryption Standard (AES) New Instructions (AES-NI) are used instead of SHA-2 for better utilization of all processing power of the CPU. This becomes especially important for Intel CPUs.

Before the evaluation of a Boolean circuit having N bit length, input from *generator* party requires the OT protocol to be applied N times. Popular frameworks for secure computations include OT protocols which use public-key operations. Instead we make use of white-box cryptography based OT protocol, which is the magnitude of order faster than other OT protocols [10], in our framework and avoid expensive public-key operations.

Both generator and evaluator parties are implemented using Message Passing Interface (MPI) library to use multiple CPUs.

4. Conclusion

In this article we have presented a general purpose secure two party computation framework offering security in semi-honest adversaries which can efficiently evaluate functions having huge circuit representations. Our compiler included in this framework can generate large circuits using smaller computational resources than similar compilers and utilizes the presence of multiple CPUs. Our evaluator also takes advantage of parallel computing resources.

We plan to make further improvements in our compiler and language to support modular programming, which will allow secure computation systems programmers to create subroutines and distribute them. As our framework is secure in the semi-honest model, we are working on adding safety against malicious adversaries.

REFERENCES

1. *A. Yao C.-C.,* How to Generate and Exchange Secrets (Extended Abstract), in 27-th Annual Symposium on Foundations of Computer Science, Toronto, 1986.

2. *Malkhi D., Nisan N., Pinkas B. and Sella Y.,* Fairplay-Secure Two-Party Computation System, in Proceedings of the 13th USENIX Security Symposium, August 9–13, 2004.

3. *Frederiksen T.K.*, Jakobsen T.P., Nielsen J. B., Nordholt P. S. and Orlandi C., MiniLEGO: Efficient Secure Two-Party Computation from General Assumptions., in EUROCRYPT, 2013.

4. *Huang Y., Katz J. and Evans D.,* Quid-Pro-Quo-tocols: Strengthening Semi-honest Protocols with Dual Execution, in IEEE Symposium on Security and Privacy, SP 2012, 21–23 May, 2012.

5. *Lindell Y., Pinkas B. and Smart N.P.,* Implementing two-party computation efficiently with security against malicious adversaries, in Security and Cryptography for Networks, Springer, 2008,/PP. 2 20.

6. *Evans D., Huang Y., Katz J. and Malka L.*, Efficient privacy-preserving biometric identification, in Proceedings of the 17-th conference Network and Distributed System Security Symposium, NDSS, 2011.

7. *Bogetoft P., Christensen D. L., Damgard I., Geisler M., Jakobsen T., Kroigaard M., Nielsen J. D., Nielsen J. B., Nielsen K., Pagter J.* and others, Secure multiparty computation goes live, in Financial Cryptography and Data Security, Springer, 2009. PP. 325–343.

8. *Crescenzo G. Di,* Private selective payment protocols, in Financial Cryptography, 2001.

9. *Even S., Goldreich O. and Lempel A.,* A Randomized Protocol for Signing Contracts, Commun. ACM, vol. 28, no. 6. PP. 637–647, 1985.

10. *Jivanyan A., Khachatryan G. and Oliynik A*. Effitient Oblivious Transfer Protocols Based on White-Box Cryptography, PERSONAL COMMUNICATION.

11. *V. Kolesnikov and Schneider T*. Improved Garbled Circuit: Free XOR Gates and Applications," in Automata, Languages and Programming, 35-th International Colloquium, 2008.

12. *Pinkas B., Schneider T., Smart N.P. and Williams S.C.* Secure Two-Party Computation Is Practical, in Advances in Cryptology-ASIACRYPT 2009, 15th International Conference, 2009.

13. *Kolesnikov V., Mohassel P. and Rosulek M*. FleXOR: Flexible garbling for XOR gates that beats free-XOR, in Advances in Cryptology-CRYPTO 2014, Springer, 2014. PP. 440–457.

14. Zakrevskii A., Minimal Realization of Partial Functions by Zhegalkin Polynomials, Avtomatika i telemekhanika, no. 5, pp. 134-140, 1996.

15. Machhout M., Zeghid M., Youssef W.E.H., Bouallegue B., Baganne A. and Tourki R., Efficient large numbers Karatsuba-Ofman multiplier designs for embedded systems, 2009.

16. Huang Y., Evans D., Katz J. and Malka L., Faster Secure Two-Party Computation Using Garbled Circuits., in USENIX Security Symposium, 2011.

ПЛАТФОРМА ОБЩЕГО НАЗНАЧЕНИЯ ДЛЯ КОНФИДЕНЦИАЛЬНЫХ ВЫЧИСЛЕНИЙ

Д. Даноян, Т. Сохакян

АННОТАЦИЯ

В статье представлена платформа общего назначения для проведения конфиденциальных вычислений между двумя получестными участниками. Платформа включает в себя собственный язык высокого уровня для описания вычисляемых функций и компилятор для построения булевой схемы, реализующий описанную функцию. Компилятор может генерировать сверхбольшие схемы, используя меньше вычислительных ресурсов по сравнению с аналогичными компиляторами и использует наличие многогопроцессорной среды. В нашей платформе используется новейший протокол забывчивой передачи, основанной на криптографии по стратегии белого ящика, что даёт нам возможность избежать дорогостоящих операций криптографии с открытым ключом.

ԸՆԴՀԱՆՈՒՐ ՆՇԱՆԱԿՈՒԹՅԱՆ ՊԼԱՏՖՈՐՄ ՆԱԽԱՏԵՍՎԱԾ ԿՈՆՖԻԴԵՆՑԻԱԼ ՀԱՇՎԱՐԿՆԵՐԻ ՀԱՄԱՐ

Դ. Դանոյան, Տ. Սոխակյան

ԱՄՓՈՓՈՒՄ

Հոդվածում ներկայացված է ընդհանուր նշանակության պլատֆորմ՝ նախատեսված կիսաազնիվ մասնակիցների կողմից իրականացվող ֆունկցիայի հաշվման համար: Պլատֆորմը ներառում է հաշվարկվող ֆունկցիաների նկարագրման համար օգտագործվող բարձր մակարդակի լեզու և նկարագրված ֆունկցիան իրացնող բուլյան սխեման կառուցող թարգմանիչ: Թարգմանիչը ունակ է նմանատիպ թարգմանիչների համեմատ ավելի քիչ հաշվարկային ռեսուրսների և բազմապրոցեսորային միջավայրի օգտագործմամբ կառուցել գերմեծ բուլյան սխեմաներ: Պլատֆորմում կիրառված են հանրային բանալիով գաղտնագրի թանկարժեք գործողությունները շրջանցող նորագույն մեթոդներ:

УДК 519.72 Поступила 16.11.2015г.

EFFECTIVE ALGORITHMS TO SUPPORT GRID FILES

G. Gevorgyan[1], **M. Manukyan**[2]

[1]*Russian-Armenian (Slavonic) University 123 Hovsep Emin St. Yerevan 0051, Armenia, grigor.gevorgyan@gmail.com*

[2]*Yerevan State University 1 Alek Manukyan St. Yerevan, 0025, Armenia, mgm@ysu.am*

SUMMARY

We present a new dynamic index structure for multidimensional data. The considered index structure is based on the extended grid file concept. Efficient algorithms for storage and access of grid file directory are proposed, in order to minimize memory usage and key retrieval complexities. Estimations of complexities for these algorithms are presented.

Keywords: grid file, algorithms, dynamic index, multidimensional data.

1. Introduction

The concept of grid files allows to effectively organize queries on multidimensionaldata [1] and can be used for efficient data cube storage in data warehouses [2, 3]. The grid file can be represented as if the space of points is partitioned in an imaginary grid. The grid lines parallel toaxes of each dimension divide the space into stripes. The number of grid lines in different dimensions may vary, and there may be different spacings between adjacent grid lines, even between lines in the same dimension.

An example of 3-dimensional grid file is presented on Figure 1. Dimensions X, Y and Z are partitioned into segments v_1, v_2, v_3 (X partitions), w_1, w_2(Y partitions) and u_1, u_2, u_3 (Z partitions). Intersections of those partitions form the *cells* of the grid file. Each of those cells contains a pointer to the data bucket, where the records corresponding to that cell are stored.

Dynamic aspects of file structures where all keys are treated symmetrically, avoiding distinction between primary and secondary keys, are studied in [5]. The

paper introduces the notions of a grid partition of the search space and of a grid directory, which are the keys to a dynamic file structure called the grid file. This file system is able to adapt to its contents under insertion and deletion operations, and thus achieves an upper bound of two disk accesses for single record retrieval. It also effciently handles range queries and partially specified queries. Several splitting and merging policies, resulting in different refinements of the grid partition are considered.

In [6] algorithms which generalize the standard single key retrieval search techniques and apply them to search of records using several keys are specfied. Two index file organization techniques, multidimensional dynamic hashing and multidimensional extendible hashing, which are multidimensional generalizations of dynamic and extendible hashing correspondingly, are specified and the average index size values for both cases, as well as their asymptotic expansions, are estimated. In particular, multidimensional extensions of linear and extensible hash tables also have been proposed in [1, 7, 8].

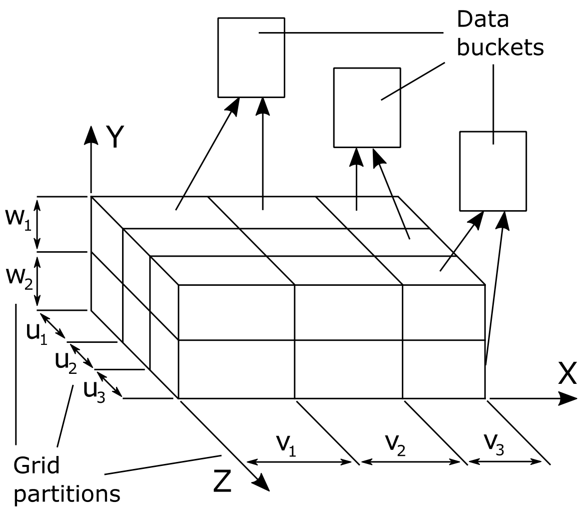


Figure 1: An example of 3-dimensional grid file

In this paper an extension of the grid file concept is proposed. Namely, we consider each stripe as a linear hash table. As a result, as opposed to grid file concept, our concept allows to increase the number of buckets more slowly. Usage of chunking technique allows us to solve the empty cells problem in grid file.

This paper is organized as follows: an approach to grid file structure modification is proposed in Section 2. Operations on grid file aredescribed in Section 3.Improvement of grid file structure aiming to reduce index file size is proposed in Section 4. Conclusions are provided in Section 5.

2. Grid file structure

One of the problems intrinsic to grid files is the problem of non-efficient memory usage by groups of cells, pointing to the same data buckets. We propose an alternative data strucure for the grid file index, aiming to avoid storage of multiple pointers to the same data buckets, as well as to maintain slow index size growth and provide reasonable costs for common operations.

Firstly, we refuse from storing the grid file as a multidimensional array. The reason for that lies within the necessity of creation of numerous new cells each time a data bucket is split, while many those cells contain duplicate pointersto the same data buckets. Instead, all cells which contain pointers to the same data buckets are grouped into chunks, represented by single memory cells with one pointer to the considered data bucket. Chunks are the main units for datainput/output, as well as are used for data clusterisation. Chunks are also used as a mechanism of struggle with empty cells problem in grid file. For each dimension, the information about how it is divided is stored in a linear scale. The data subspace, corresponding to a particular element of that scale, is called a stripe, and is represented by an array of pointers to corresponding chunks.

Secondly, we consider each stripe as a linear hash table, allowing usage of overflow blocks to reduce the number of chunk partitions. The number of overflow blocks may be different for different chunks, however we ensure that for any stripe the average number of overflow blocks for the chunks of that stripe is not greater than one. This allows us to significantly reduce the total number of chunks, while guaranteeing not more than two disk operations for data access in average.

An example of 2-dimensional grid file is presented on Figure 2. It contains five chunks, each pointing to corresponding data bucket. Two chunks use overflow blocks. The solid lines represent borders between chunks, and dashed lines mean imaginary space divisions.

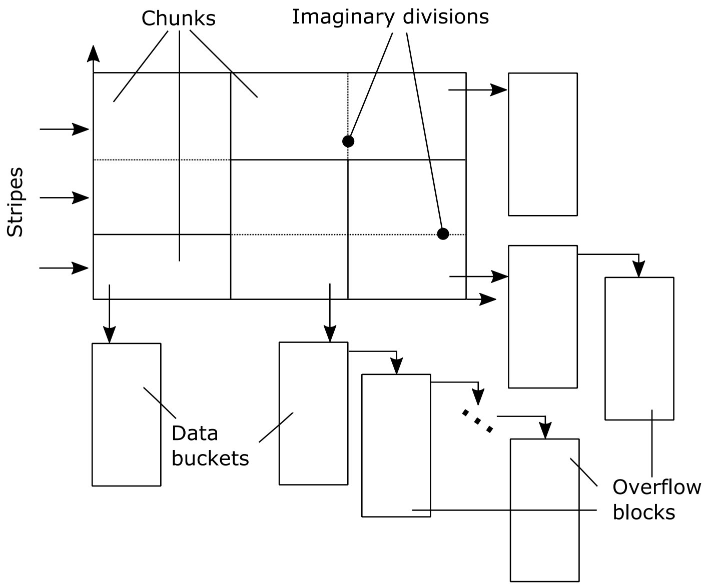


Figure 2: An example of 2-dimensional grid file

2.1. Formalization

Let us formally represent a grid file as a triple $F = \langle D, S, C \rangle$ where D is the set of dimensions, S is the set of stripes, and C is the set of cells. Each stripe belongs to exactly one dimension. If stripe s belongs to the dimension d, we say that s is a stripe of dimension d and denote it as $s \in d$. We call a set of cells $A \subseteq C$ a grid file chunk (or just chunk), if the divisions between those cells are imaginary, and they all refer to the same bucket.

Observation 1. If only split operations are performed on the grid file, each grid file chunk has a shape of n-dimensional parallelepiped.

Proof. If we perform a split operation on a parallelepiped-shaped chunk, it shall split into two parallelepiped-shaped chunks. Since the grid file originally has a form of n-dimensional parallelepiped, and every chunk is obtained by consecutive applications of split operation, hence each chunk will be parallelepiped-shaped at each point of time.

2.2. Several concepts and their estimations

In this paragraph we investigate several grid file concepts and derive estimations important for further material. Analysis are done under the assumption that all dimensions of considered grid file data domain are independent and equivalent. Let us denote the number of dimensions as n and the average number of splits performed in one dimension as m.

Number of cells. Since each of n dimensions is divided into m parts in average, there exist $\Theta(m^n)$ cells in average.

Number of stripes in one particular dimension is equal to number of splits towards that dimension $\Theta(m)$ in average.

Total number of stripes is hence $\Theta(nm)$ in average.

Total number of chunks. One new chunk is created as a result of a split operation, hereby the number of chunks is in direct proportion with the number of splits is $\Theta(nm)$.

Number of cells per chunk in average is the total number of cells divided by number of chunks is

$$\Theta(\frac{m^n}{mn})$$

Average length of chunk side. Using Observation 1, and assuming that the average shape of a chunk is an n-dimensional cube, the average length of its side is

$$\Theta\left(\sqrt[n]{\frac{m^n}{mn}}\right) = \Theta\left(\frac{m}{\sqrt[n]{mn}}\right)$$

Average number of chunks crossed by a single stripe. A stripe has an average length of m cells in all of its n-1 dimensions. Since the average length of a chunk side is

$$\Theta\left(\frac{m}{\sqrt[n]{mn}}\right)$$

a stripe will cross

$$\Theta\left(\left(\frac{m}{\frac{m}{\sqrt[n]{mn}}}\right)^{n-1}\right) = \Theta\left(\left(\sqrt[n]{mn}\right)^{n-1}\right)$$

For simplicity of reasoning we shall weaken this estimation to $O(mn)$.

The following table contains summary of the estimated values:

Value	**Assessment**
Number of dimensions	$\Theta(n)$
Number of divisions per dimension	$\Theta(m)$
Total number of cells	$\Theta(m^n)$
Number of stripes per dimension	$\Theta(m)$
Total number of stripes	$\Theta(nm)$
Total number of chunks	$\Theta(nm)$
Number of cells per segment	$\Theta(\frac{m^n}{mn})$
Average length of segment side	$\Theta\left(\frac{m}{\sqrt[n]{mn}}\right)$
Average number of segments crossed by stripe	$O(mn)$

Table 1: Estimated average values

2.3. Directory size

Since each of the $\Theta(nm)$ stripes crosses $O(mn)$ chunks in average, the total number of stored pointers will be equal to $O(n^2m^2)$. Also each chunk has one pointerto data bucket, $\Theta(nm)$ pointersin total. Hereby, the grid file directory size in this case is $\boldsymbol{O(n^2m^2)}$.

3. Grid file operations

In this section we review basic operations performed with usage of grid file. We distinguish two aspects of operations: index modification expences (estimated in terms of Big-O notation) and operation costs expressed in disk operations.

3.1. Index modifications

Insert. To insert a value we first locate the chunk it belongs to by given coordinates, then read the corresponding data bucket and perform insertion. Since we consider each stripe as a linear hash table, in case of insufficient space it may be possible to add an overflow block. However, if after insertion there exists a stripe for which the average number of overflow per chunk exceeds 1, we need to split current chunk into two parts, and reorganize corresponding data buckets.

The following pseudocode illustrates the insertion procedure.

Algorithm 1. Insertion

```
Insert(v):
  in: v – value to insert
  c ← FindChunk(v)                          ▷ Get corresponding chunk
  b ← LoadLastBlock(c)                      ▷ Load the last block of that chunk
  if not Full(b) or CanOverflow(c) then     ▷ Check if insertion is possible
      AddRecordToBlock(v, b)                ▷ Perform insertion
  else
      Split(c)                              ▷ Perform a split
      Insert(v)                             ▷ Try insertion again
  end if
```

The AddRecordToBlock function used above tries to insert provided value into the given block. If it is not possible due to occupancy of the block, it craetes an overflow block and inserts value there. It is illustrated by pseudocode below:

Algorithm 2. Addition of a record to block

```
AddRecordToBlock(v, b):
  in:v - value to insert, b - block to try insertion
  out: block where the value was actually inserted
  if Full(b) then                   ▷ Add an overflow block if necessary
      AddOverflowBlock(b)
      b ← LoadNextBlock(b)
  end if
      WriteRecordToBlock(v, b)      ▷ Add record to block and save to disk
      SaveBlock(b)
  returnb
```

Split. Consider a chunk$c$ which is needed to be split. Firstly, we choose the dimension in which the split shall be performed. For that purpose we choose some dimension d which currently contains the least number of stripes. Secondly, we choose some coordinate d_0 in dimension $d$which will define the dividing hyperplane. Chunk c then is split into two chunks c_1 and $c_2$with all points from c with d coordinate less than d_0 belonging to a_1, and the rest point belonging to a_2. The

coordinate $d_0$is chosen in such a way that the numbers of points in the resulting chunks are as close as possible. This can be achieved by storing additional statistical data for each chunk – most frequent values and their quantities (*mfv*) and total number and average value of infrequent values (*oav*). As a result of split, some stripe s in dimension d is also divided into two stripes $s_1$and s_2. Pointers to the newly created chunks should be added to pointer lists of allstripes crossing them. Split operation pseudocode is presented as Algorithm 3:

Algorithm 3.Splitting a chunk

Split(c):
in: c – chunk to split
Find the split dimension d and split coordinate d_0 in it
Initialize two new chunks c_1 and c_2
Load first blocks of chunks c, c_1 and c_2 as b, b_1 and b_2 correspondingly
repeat
for all $r \in b$ **do**
if $\Pi_d(r) \leq d_0$ **then** ▷ Redistribute all records from
$b_1 \leftarrow$ Add Record To Block(r, b_1) ▷ old chunk to the new ones
else
$b_2 \leftarrow$ Add Record To Block(r, b_2)
end if
end for
$b \leftarrow$ Load Next Block(b)
until Exists(b)
$s \leftarrow$ Find Stripe(d, d_0)
Split stripe s by coordinate d_0 into two new empty stripes s_1 and s_2
for all $c_0 \in$ Stripe Chunks(s) **do**
Add chunk c_0 in to strupes $s_1$and s_2
end for
Add chunks c_1, c_2 to stripes s_1, s_2 correspondingly
for all $s \in$ Stripes Of Chunk(c) **do**
Add chunks $c_1$and c_2 to stripe s
end for
Remove Chunk(c) ▷ Remove the old chunk pointer

As an example, let us consider a 2-dimensional grid file with coordinate axis X and Y. Assume that each data block has capacity enough to contain not more than

5 records. Analogously to linear hash tables, we shall ensure that in each stripe the ratio of average number of records in a block to block size does not exceed some constant value [1]. We shall use the value of 80% for this purpose. Let us denote number of records in a stripe as r and number of chunks in a stripe as c. From the above condition we get that$r/c \leq 4$ inequality must take place. Let us also denote r/c ratio as k.

At first, we have one chunk A_1, containing two points (1; 1) and (5; 5). We also have one horizontal stripe S_h^1 and one vertical stripe S_v^1. Then we insert two points (1; 5) and (4; 1). After that, for each stripe $r = 4$; $c = 1$ and $k = 4$, so we can proceed without splitting the chunk. Since there is enough space in data bucket, the records will be just added there.

This process is illustrated in Figure 3.

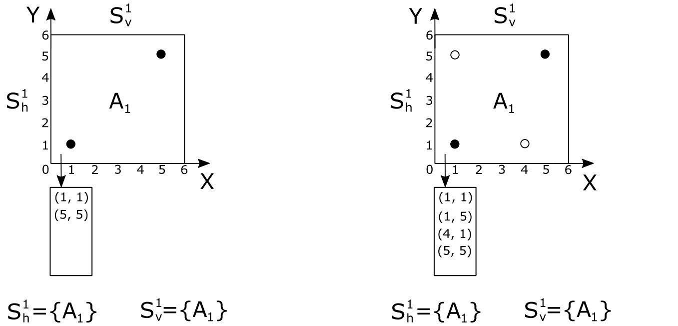
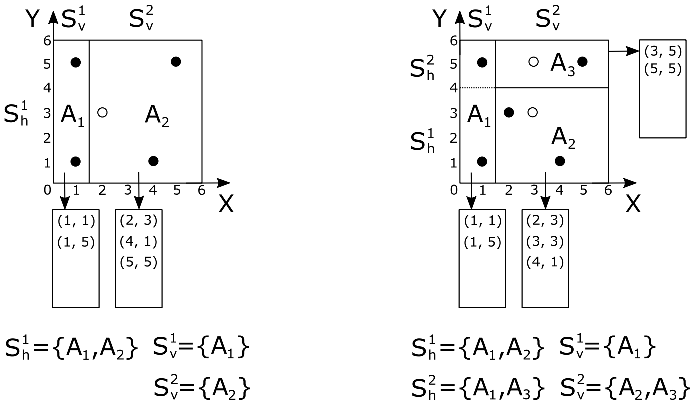


Figure 3: A simple insert operation



(a) Inserted (2, 3) (b) Inserted (3, 3) and (3, 5)

Figure 4: Split operations

Let us insert another point (2; 3). If we insert it into chunk A_1we shall get $r = 5$; $c = 1$ and $k = 5 > 4$, so we have to split the chunk. After splitting it vertically with $x = 1.5$ line, we obtain two chunks $A_1$and A_2, as well as anoter vertical stripe S_v^2. Now for stripe S_h^1 parameter c equals to 2, because S_h^1 now consists of two chunks, and $k = 2.5$ allows us to proceed with insertion.

It is fine to insert another point (3; 3) into chunk A_2, but after adding one more point (3; 5) we face a necessity to split the chunk A_2, because this time $k = 5 > 4$ for stripe S_v^2. We split it horizontally with line $y = 4$. That line also crosses chunk A_1, but we do not split it too. The result of above operations is illustrated on Figure 4. We represent the imaginary split of chunk A_1 with a dashed line.

Finally, let us insert 3 more points: (3; 2), (5; 1) and (5; 3). There is enough space for (3; 2) and (5; 1) in chunk A_2, however the third point (5; 3) does not fit there because the corresponding bucket is already full. However, for both stripes S_h^1 and S_v^2 crossing the chunk A_2 we get $r = 8$; $c = 2$ and $k = 4$, so we can avoid splitting the chunk (and increasing directory size) by using an overflow block. The result of such insertion is presented on Figure 5.

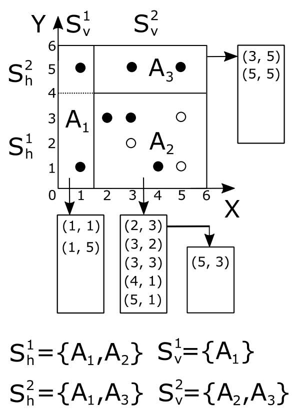


Figure 5: Overflow block usage

Complexity. A single split operation increments the number of chunks and the number of stripes. Per Table 1, the new stripe pointer list will contain $\Theta(nm)$ elements in average. Also, $\Theta(nm)$ pointers will be added into pointer lists of all other dimensions. Hence, split operation will increase directory size by $\Theta(nm)$.

Delete. To delete a value, as when inserting, we first find the chunk it belongs to and read the corresponding data block. After deletion, the data block may become empty, meaning the considered chunk does not contain any more points. In

such cases, we can merge this chunk with some neighbour chunk. Even if the data block does not become empty, it may still be possible to merge the chunk with a neighbor chunk, if the condition of having not more than one overflow block per chunk in average is satisfied. The dimension towards which to perform the merge is again chosen in such a way that the number of stripes in different dimensions differs not more than by 1.

Merge. Consider two chunks a_1 and a_2 which have a common boundary in dimension d and are being merged into a single chunk a. For any stripe s if its pointer list contains either a pointer to a_1 or a_2, those should be replaced with a pointer to a instead. There is a possibility that after this operation an empty stripe arises, and has to be removed.

Merging algorithm is illustrated in the following pseudocode:

Algorithm 4. Merging two chunks

```
Merge (c1, c2):
  in: c1, c2 – chunks to merge
  1:  d ← Boundary Dimension (c1, c2)
  2:  Load first blocks of chunks c1, c2 as b1, b2 correspondingly
  4:  repeat
  3:  for all r in b2 do
  4:   b1 ← Add Record To Block (r, b1)
  5:  end for
  6: b2 ← Load Next Block (c2)
  7:   until Exists (b2)
  8: for all s ∈ Stripes Of Chunk (c2) do
  9: Add chunk c1 to stripe s
  10: Remove chunk c2 from stripe s
  11: end for
  12: return c1
```

Complexity. A single merge operation will remove one chunk and the number of pointers equal to the number of stripes crossing the merged chunks – $\boldsymbol{\Theta(nm)}$ in average.

Single key retrieval. In order to find all records for a key k from dimension d we first find the stripe s which k belongs to, and then traverse over all the chunks which are accessible from pointer list of s. Since all of these chunks have to be con-

sidered, and no other chunks may contain records matching the key k, only necessary and sufficient chunks will be traversed. The average number of such chunks is $O(nm)$ according to Table 1. Assuming that the required stripe s can be found in $O(\log m)$ time, the time complexity of a single key retrieval operation is $\boldsymbol{O(nm \log m)}$.

Here is the single key retrieval pseudocode:

Algorithm 5. Finding values by a single key

Find(k, d):
in: k – lookup key, d – lookup dimension
out: set of all records matching the lookup criteria
$result \leftarrow \emptyset$
$s \leftarrow$ FindStripe(d, k)
for all $c \in$Chunks Of Stripe(s) **do**
$b \leftarrow$ Load First Block(c)
repeat
for all $r \in b$ **do**
if $\Pi_d(r) = k$ **then**
$result \leftarrow result \cup \{r\}$
end if
end for
$b \leftarrow$ Load Next Block(b)
until Exists(b)
end for
return $result$

Multiple keys retrieval. Multiple keys retrieval is performed in two steps. First, a single key retrieval procedure is called for each of the keys, each one separately retrieving a set of referable chunks. These sets are then intersected, and only buckets corresponding to chunks in the intersection are being considered. Time complexity to perform a query using k keys is thus $\boldsymbol{O(knm \log m)}$.

Algorithm 6. Finding values by multiple keys

Find(S):
in: S – set of $\langle key, dimension \rangle$pairs

out: set of all records matching the lookup criteria
1: $result \leftarrow \emptyset$
2: **for all** $\langle k, d \rangle \in S$ **do** ▷ Lookup by all keys separately
3: $result \leftarrow result \cap$ Find(k, d) ▷ and intersect the results
4: **end for**
5: **return** $result$

3.2. Disk operations

Point lookup. To find a point provided its coordinates, we first find the chunk it belongs to in the grid file, then read the corresponding data bucket. This requires at most two disk operations in average, since the average number of overflow blocks per bucket is not more than one. Additionally, if an update operation has to be performed, a write opera tion is needed. Besides, adding/deleting a record can result in bucket overflow or exhaustion, rising a necessity to perform a split or merge operation correspondingly, resulting in another disk operation.

Queries with several coordinates and value ranges. We shall traverse over all chunks, belonging to intersection of the stripes corresponding to the provided coordinates, or to the grid file subspace matching the provided ranges. Number of disk operations equals to the number of considered chunks times two (due to overflow blocks).

Closest object lookup. To find the point, closest to the provided one, we first find the chunk it belongs to, and check corresponding data buckets. It may happen that, however, the nearest point belongs to a neighbor chunk, and it needs to be considered as well. In worst case, we shall need to visit all n neighbor chunks. Since reading data belonging to one chunk requires not more than 2 disk operations in average, in worse case we shall need to perform $2(n+1)$ disk operations. Note that necessity to add/remove a record and potential split/merge operations will also increase the number of disk operations.

4. Improvement of the grid file structure

In this section a grid file structure modification is proposed, allowing to reduce directory size from $O(n^2m^2)$ to $O(n^2m)$. To achieve this goal we reorganize pointers storage, allowing chunks to refer to each other:

Definition 1. *Let there be a total order* $<$ *defined on the set of chunks. Let us also denote the projection of chunk* A *to dimension* d *as* $\Pi_d(A)$. *The set of pointers* E *is defined as*:

1. *For each pair of chunks A and B s.t. $A < B$ and a dimension d exists s.t. $\Pi_d(A) \subseteq \Pi_d(B)$, and no chunk C exists s.t. $A < C < B$ and $\Pi_d(A) \subseteq \Pi_d(C), \Pi_d(C) \subseteq \Pi_d(B)$, there exists a pointer (A, B) in E. Let us call such pointer a pointer of dimension d.*
2. *For each chunk A and stripe s of dimension d, if there exists no pointer of dimension d incoming into A, then there exists a pointer (s, A) in E.*

4.1. Directory size

Consider the sequence of chunks $A_1, A_2, \dots, A_k$ $(A_i < A_j \Leftrightarrow i < j)$ crossed by a stripe s of dimension$d$. For each pair of chunks A_i and A_j $(i < j)$, let us consider their projections $\Pi_d(A_i)$ and $\Pi_d(A_j)$. These projections intersect, because both segments are crossed by the stripe s. There are four possible relative positions of $\Pi_d(A_i)$ and $\Pi_d(A_j)$, and only one of them meets the condition of $\Pi_d(A_i) \subseteq \Pi_d(A_j)$. Based on the assumption that all of these four cases are equiprobable for each pair of chunks, we can say that the probability of having a pointer between from chunk A_i to chunk A_j is:

$$P\left(\left(A_i, A_j\right) \in E\right) = \frac{1}{4}.$$

Let us calculate the expected number of pointers $\left(s, A_j\right)$. For each chunk A_j such a pointer exists, if there is no other chunk A_i s.t. $A_i < A_j$ and $\Pi_d(a_i) \subseteq \Pi_d(A_j)$.

Since

$$\forall\, i \in [1, j-1], P\left(\left(A_i, A_j\right) \notin E\right) = \frac{3}{4}$$

then

$$P\left(\forall\, i \in [1, j-1], \left(A_i, A_j\right) \notin E\right) = \left(\frac{3}{4}\right)^{j-1}$$

The expected number of such pointers is thus:

$$\sum_{j=1}^{k} \left(\frac{3}{4}\right)^{j-1} = 4\left(1 - \left(\frac{3}{4}\right)^{k}\right) < 4$$

So the total number of pointers stored in each stripe's pointer list does not exceed 4 in average. Taking into account that each chunk A has n stored pointers (one per each dimension), and there are $\Theta(nm)$ chunks in average, we can estimate the total number of pointers – as well as directory size – as:

$$|E| = O(4nm + n^2 m) = O(n^2 m).$$

4.2. Comparison of directory size

Our approach proposesseveral optimizations to reduce the directory size. One is considering each stripe as a linear hash table, allowing not more than one overflow block per bucket in average. We also use the satistical values *mfv* and *oav* during chunk splitting, described in Section 3.1, aiming for uniform filling of the chunks. Another optimization is usage of chunking technique to solve the problem of empty cells in grid file. In [6] two main techniques for grid file organization, multidimensional dynamic hashing (MDH) and multidimensional extendible hashing (MEH), are desribed and estimates of directory sizes for both cases are provided: $\boldsymbol{O}\left(\boldsymbol{r}^{\mathbf{1}+\frac{\mathbf{1}}{\boldsymbol{s}}}\right)$ and $\boldsymbol{O}\left(\boldsymbol{r}^{\mathbf{1}+\frac{\boldsymbol{n}-\mathbf{1}}{\boldsymbol{ns}-\mathbf{1}}}\right)$ correspondingly, where r is the number of records, s is the block size and n is the number of dimensions. It should be noted that we consider the case of uniform distributions.

For comparison let us express the directory size in case of our approach using these values.Since we allow each chunk to have one overflow block in average, we can, without loss of generality, assume that each of the overflow blocks will be half-full in average, meaning that we shall store $1.5s$ records per chunk in average. Hereby we can conclude that it will be required to have $\frac{r}{1.5s}$ chunks to store all$r$ records, which is equivalent to $\boldsymbol{\Theta(nm)}$ according to Table 1. Hereby, according to Section 4.1, our directory size can be estimated as $\boldsymbol{O}\left(\frac{2nr}{3s}\right)$. Compared to MDH and MEH techniques, directory size in our approach is $\boldsymbol{O}\left(\frac{\mathbf{3}\boldsymbol{sr}^{\frac{\mathbf{1}}{\boldsymbol{s}}}}{\mathbf{2}\boldsymbol{n}}\right)$ and $\boldsymbol{O}\left(\frac{\mathbf{3}\boldsymbol{sr}^{\frac{\boldsymbol{n}-\mathbf{1}}{\boldsymbol{ns}-\mathbf{1}}}}{\mathbf{2}\boldsymbol{n}}\right)$ times smaller correspondingly.

5. Conclusion

In this paper a new dynamic indexing scheme for effective data management is proposed. Our approach to such index structure is based on the grid file concept. This concept is extended within our approach. Namely, we consider each stripe as a linear hash table. As a result, the number of overflow blocks for each bucket per stripe is less than one in average, and the growth of buckets quantity is slow. We propose a modification of grid file structure, providing analysis of memory usage and costs of add, remove, split, merge, and lookup operations. Usage of chunking technique allows us to solve the empty cells problem intrinsic to grid files. Efficient algorithms for storage and access of grid file directory are proposed and estimations of complexities for these algorithms are presented.

REFERENCES

1. *Garcia-Molina H., Ullman J/ and Widom J.,* Database Systems: The Complete Book. Prentice Hall, USA, 2009.
2. *Whang K.-Y. and Krishnamurthy R.,* The multilevel grid file – a dynamic hierarchical multidimensional file structure. In DASFAA Conference, pages 449–459, 1991.
3. *Luo C., Hou W.C., Wang C. F., Want H. and Yu X.,*Grid file for efficient data cube storage. Computers and their Applications, pages 424–429, 2006.
4. *Karayannidis N.N.,* Storage Structures, QueryProcessing, and Implementation of On-Line AnalyticalProcessing Systems (Ph.D. Thesis). Athens, 2003.
5. *Nievergelt J. and Hinterberger H.,*The grid file: Anadaptable, symmetric, multikey file structure. ACMTr ansactions on Database Systems, 9 (1):38–71, March 1984.
6. *Regnier M.,* Analysis of grid file algorithms. BIT, 25 (2):335–358, 1985.
7. *Otoo E.J.,* A mapping function for the directory of amultidimensional extendible hashing. In 10th International Conference on VLDB, pages 493–506. Singapore, Aug 1984.
8. *Otoo E. J.,* A multidimensional digital hashing schemefor files with composite keys. In ACM SIGMOD International Conference on Management of Data, pages 214–229. USA, 1985.
9. *Papadopoulos A.N., Manolopoulos Y., Theodoridis Y. and Tsoras V.,* Grid file (and family).In Encyclopedia of Database Systems. PP. 1279–1282, 2009.
10. *Sharma S., Tim U. S., Wong J., Gadia S. and Sharma S.,* A brief review on leading big data models. In Data Science Journal, December 2014.

ЭФФЕКТИВНЫЕ АЛГОРИТМЫ ДЛЯ ПОДДЕРЖКИ СЕТОЧНЫХ ФАЙЛОВ

Г. Геворгян, М. Манукян

АННОТАЦИЯ

Предложена динамическая структура индекса для многомерных данных. Рассматриваемая структура индекса основана на понятии сеточного файла. С целью минимизации объема используемой памяти и времени поиска по ключу предлагаются эффективные алгоритмы хранения и доступа к директории индекса. Приводятся оценки сложности этих алгоритмов.

ՑԱՆՑԱՅԻՆ ՖԱՅԼԵՐԻ ԱՋԱԿՑՄԱՆ ԷՖԵՏԿՏԻՎ ԱԼԳՈՐԻԹՄՆԵՐ

Գ. Գևորգյան, Մ. Մանուկյան

ԱՄՓՈՓՈՒՄ

Առաջարկվել է բազմաչափ տվյալների ինդեքսավորման նոր դինամիկ կառուցվածք: Դիտարկվող կառուցվածքը հենվում է ցանցային ֆայլի հասկացողության վրա: Օգտագործվող հիշողության ծավալի և բանալիով որոնման ժամանակի մինիմիզացիայի նպատակով առաջարկվել են ինդեքսի պահպաման և օգտագործման էֆեկտիվ ալգորիթմներ: Բերվում են այդ ալգորիթմների բարդության գնահատականները:

УДК 004.056.55 Поступила 13.11.2015 г.

REVIEW OF SEARCHABLE ENCRYPTION ALGORITHMS

M. Hovsepyan

Russian-Armenian University Department of System Programming
Republic of Armenia, Yerevan, 0051
hovsepyan.mihran@gmail.com

SUMMARY

Searchable encryption allows the user to store his data in untrusted environment such as public cloud storages in encrypted form but still be able to access the data via search. Meantime the storage provider cannot learn neither the data nor even the search queries. The importance of such functionality raised with the wide adoption of public cloud storages such are Dropbox or Google Drive and this discipline gained high attention from research community. This paper provides a generic survey for practical searchable encryption schemes along with light analysis of advantages and disadvantages of each schema.

Keyword: secure, search, cryptography, cloud storage.

1. Introduction

Searchable encryption is a technique that allows a client to store documents on a server in encrypted form without sacrificing the user experience of accessing the data. This technique has significant importance in this cloud-driven era and the area of searchable encryption has been identified by Defense Advanced Research Projects Agency (DARPA) as one of the most important technical advances that can be used to balance the need for both privacy and national security in information aggregation systems [1].

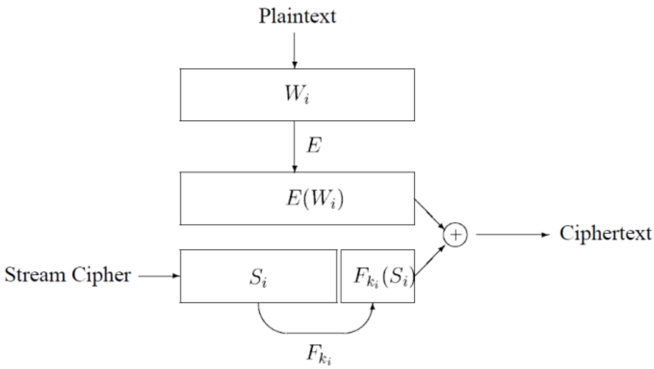
In recent years many practical schemes have been developed for performing searchable encryption with optimal search time and efficient memory consumption with reasonable security leakage and for different settings. In the symmetric searchable encryption domain, the indexing and the retrieval are performed by the same client. The symmetric searchable encryption algorithms can be either static, which allows indexing the data once and then perform search functionality, or dynamic allowing updating the index upon changes of the data. In the public-key searchable

encryption setting the storage can be done by different users but the retrieval is performed by the only client. In the multi-key searchable encryption the storage and retrieval can be done by different users. In this setting the access graph defines the retrieval permissions for different users. We review the most efficient and practical searchable encryption algorithms from three different domains.

2. Symmetric Searchable Encryption (SSE) Schemes

The most practical symmetric searchable encryption schemes are based on the so-called secure index approach. In an index-based SSE scheme [4, 5, 6, 7, 8, 9, 11] the encryption algorithm takes as input the sequence of n files $f = (f_1, \ldots, f_n)$ and outputs an encrypted index and a sequence of n ciphertexts $c = (c_1, \ldots, c_n)$, one for each file. All known constructions construct c encrypting the files f using any symmetric encryption scheme, i.e., the file encryption does not depend on any unusual properties of the encryption scheme. To search for a keyword w the client generates a search token t_w and given t_w the server can find the identifiers I_w of the files that contain w.

But the idea of secure index was not appeared at once. In the paper by D. X. Song, D. Wagner, and A. Perrig, "Practical techniques for searches on encrypted data" [1] the problem of searchable encryption has been explicitly considered for the first time and a scheme with search time that is linear in the size of data collection has been presented. The basic idea was to encrypt the file word by word in a specific manner shown in the graphic below.



As can be seen the key owner can easily construct the ciphertext for the encrypted words and search it in the encrypted files. Their construction supports insertions/deletions of files in a straightforward way as each file was encrypted word by

word in a specific manner. However this construction has significant information leakage as well as non-practical characteristics.

Symmetric searchable encryption can be achieved in its full generality and with optimal security using the work of Ostrovsky and Goldreich on oblivious RAMs [2,3]. More precisely, using these techniques, any type of search query can be achieved (e.g., conjunctions or disjunctions of keywords) without leaking any information to the server, not even the "access pattern" (i.e., which documents contain the keyword). This strong privacy guarantee, however, comes at the cost of a logarithmic (in the number of documents) number of rounds of interaction for each read and write.

Next Goh [4] proposed a dynamic solution for SSE introducing the concept of Secure Index. This secure index allows a user to search for an encrypted document that is containing a keyword without decrypting the document. A Bloom Filter is used as a document index to keep track of each of the unique words. Before each of the unique keywords is indexed and stored into Bloom filter objects, those unique keywords have to go through a pseudorandom function twice. The purpose of doing so is to make sure that for each of two or more documents, if they contain the same keyword the code word will represent them differently. The solution requires linear search time, and because the use of Bloom filters can result in false positives, the search may return indexes already deleted from the index. Nevertheless, the Secure Index is used as the basis of all upcoming constructions. Next we present the latest most efficient and practical symmetric searchable encryption algorithms.

2.1. Highly Scalable Static SSE Scheme with Support for Boolean Queries

The next static searchable encryption scheme [6] has been designed specially to efficiently support not only search of single keyword but also their Boolean combinations particularly multi-keyword searches. The specifications of the main functions are given bellow:

- $\mathrm{Gen}(1^k)$: In this initialization step the data owner (client) using a random bitstring (password) generates three k-bit uniformly random keys K_S, K_X, K_T. Also in this step we choose a keyed hash function $F: \{0,1\}^k \times \{0,1\}^* \to \{0,1\}^k$ (for example HMAC or CMAC).

– $\text{Enc}(K_S, K_X, K_T, f)$: For the given set of files f client generates secure index using generated in the first step keys and stores them into the untrusted 3rd party data storage (server).

a. Parse the file collection f as $W = \left(w_i, \text{Fid}_{w_i}\right)_{i=1}^{D}$ where D is the count of all different keywords in all files and Fid_{w_i} is the set of file ids which contain the word w_i.

b. Initialize XSet empty set and A empty dictionary.

c. For each keyword w from W build A dictionary and XSet as follows.

1. Initialize t to be an empty list of k-bit strings.
2. Set $\text{K}_\text{e} \leftarrow \text{F}(\text{K}_\text{S}, \text{w})$, $\text{xtrap} \leftarrow \text{F}(\text{K}_\text{X}, \text{w})$, and $\text{stag} \leftarrow \text{F}(\text{K}_\text{T}, \text{w})$
3. For all file indexes $\text{fid} \in \text{Fid}_w$ which contain the keyword w:
   1. Compute $e \leftarrow \text{Enc}(K_e, \text{fid})$ and append e to t.
   2. Compute $\text{xtag} \leftarrow \text{F}(\text{xtrap}, \text{fid})$ and add xtag to XSet.
   3. $\text{A}[\text{stag}] \leftarrow t$.

d. Store the encrypted index $(A, \text{ XSet})$ along with the encrypted files into the untrusted server.

– $\text{Trpdr}(K_T, K_S, K_x, w_1, w_2, \dots, w_\text{n})$: For searching some Boolean combination of keywords the client generates special tokens (trapdoor) using the secret keys K_S, K_X, K_T as follows:

a. $K_e \leftarrow F(K_S, w_1)$, $\text{stag} \leftarrow F(K_T, w_1)$,

b. For each $i = 2, \dots, n$ set $\text{xtrap}_i \leftarrow F(K_X, w_i)$

c. Outputs the $(\text{stag}, K_e, \text{xtrap}_2, \dots, \text{xtrap}_n)$ as the trapdoor.

– $\text{Search}\left((\text{stag}, K_e, \text{xtrap}_2, \dots, \text{xtrap}_n), \text{A}, \text{XSet}\right)$: The untrusted server gets the trapdoor from the client and performs the following actions:

a. $t \leftarrow A[\text{stag}]$.

b. For each ciphertext e in t, it computes:

1. $\text{fid} \leftarrow \text{Dec}(K_e, e)$
2. For all i from 2 to n check whether fid file contain w_i by checking if $F(\text{xtrap}_i, \text{fid}) \in \text{XSet}$. Return fid if it contains only the required keywords.

From the description of the index and query construction and search algorithm it is clear that using this approach any Boolean combination of keywords can be quickly searched after converting it to the disjunctive normal form and creating trapdoors for each conjunction.

2.2. Dynamic Multi-User Symmetric SSE

In the real world the data is always dynamic meaning that files can be added or deleted or even the content of each unique file may be changed in the data collection. In this scenario it is desirable to index the data with dynamic methods allowing to efficiently updating the secure index without requiring of re-indexing of the whole database at each update. Below we review one of the most efficient and popular dynamic searchable encryption schemes.

In [7] a practical multi-user SSE is presented which allows dynamically add and delete files from the secure index in an efficient way. The construction is based on the inverted index approach. The scheme is a tuple of 9 algorithms (Gen, Enc, SearchToken, AddToken, DelToken, Search, Add, Del, Dec). Here we describe only the index construction and search methods referring the reader to the full paper for details on how to update the secure index.

– $\mathrm{Gen}(1^k)$: Select uniformly random four k-bit keys $K := (\mathrm{K}_1, \mathrm{K}_2, \mathrm{K}_3, \mathrm{K}_4)$.
– $\mathrm{Enc}(K, f)$: Using index generation keys K and the user's document collection $f = (f_1, \ldots, f_n)$ generates an encrypted index γ and encrypted documents $c = (c_1, \ldots, c_n)$ in the following way:
  a. Let $\mathrm{A_s}$ and $\mathrm{A_d}$ be arrays of size $\frac{|\boldsymbol{c}|}{8} + z$ and let $\mathrm{T_s}$ and $\mathrm{T_d}$ be dictionary of size $\#W$ and $\#\mathbf{f}$, respectively. **0** Is a $(\log \# A_s)$-length string of 0's and **free** is a word not in W.
  b. for each word $w \in W$:
    1. Create a list L_w of $\#\mathbf{f}_w$ nodes $\left(N_1, \ldots, N_{\#\mathbf{f}_w}\right)$ stored at random locations in the search array $\mathrm{A_s}$ and defined as:
$$\mathrm{N_i} := (\langle \mathrm{id_i}, \mathrm{addr_s}(\mathrm{N_{i+1}}) \rangle \oplus \mathrm{H_1}(\mathrm{K_w}, \mathrm{r_i});\ \mathrm{r_i})$$
       where $\mathrm{id_i}$ is the ID of the i-th file in $\mathbf{f}_w$, r_i is a k-bit string generated uniformly at random. $\mathrm{K_w} := \mathrm{P}_{\mathrm{K}_3}(\mathrm{w})$ and $\mathrm{addr_s}\left(\mathrm{N}_{\#\mathbf{f}_\mathrm{w}+1}\right) = \mathbf{0}.$
    2. Store a pointer to the first node of L_w in the search table by setting
$$\mathrm{T_s}\left[\mathrm{F}_{\mathrm{K}_1}(\mathrm{w})\right] := \langle \mathrm{addr_s}(\mathrm{N_1}), \mathrm{addr_d}(\mathrm{N_1^*}) \rangle \oplus \mathrm{G}_{\mathrm{K}_2}(\mathrm{w})$$
       where N^* is the dual of N, i.e., the node in $\mathrm{A_d}$ whose fourth entry points to $\mathrm{N_1}$ in $\mathrm{A_s}$.
  c. for each file f in **f**:
    1. Create a list L_f of $\#\overline{f}$ dual notes $\left(D_1, \ldots, D_{\#\overline{f}}\right)$ stored at random locations in the deletion array A_d and defined as follows: each entry D_i is associated with a word w, and hence a node N in L_w . Let N_{+1} be the

node following N in L_w and N_{-1} the node previous to N in L_w. Then, define D_i as follows:

$$D_i := \left(\langle \begin{matrix} \mathrm{addr_d(D_{i+1}), addr_d(N^*_{-1}), addr_d(N^*_{+1}), addr_s(N),} \\ \mathrm{addr_s(N_{-1}), addr_s(N_{+1}), F_{K_1}(w)} \end{matrix} \rangle \oplus \mathrm{H_2(K_f, r_i^{'}); \ r_i^{'}} \right)$$

where $r_i^{'}$ is a k-bit string generated uniformly at random. $\mathrm{K_f} := \mathrm{L_{K_7}(f)}$, and $\mathrm{addr_d}\left(\mathrm{D}_{\#\bar{f}+1}\right) = \mathbf{0}.$

2. store a pointer to the first node of L_f in the search table by setting

$$\mathrm{T_d}\left[\mathrm{R_{K_5}(f)}\right] := \langle \mathrm{addr_s(D_1)} \rangle \oplus \mathrm{M_{K_6}(f)}$$

d. Create an unencrypted free list $\mathrm{L_{free}}$ by choosing z unused cells at random in A_s and in A_d. Let $(F_1, \dots, F_z)$ and $(F_1^{'}, \dots, F_z^{'})$ be the free nodes in A_s and A_d, respectively. Set

$$T_s[\mathrm{free}] := \langle \mathrm{addr_s(F_z)}, \mathbf{0}^{\log \#A} \rangle$$

and for $z \geq i \geq 1$ set

$$\mathrm{A_s[addr_s(F_i)]} := \langle \mathbf{0}^{\log \#f}, \mathrm{addr_s}\,(\mathrm{F_{i-1}}), \mathrm{addr_d}\left(\mathrm{F_i^{'}}\right) \rangle$$

where $\mathrm{addr_s}\,(\mathrm{F_0}) = \mathbf{0}^{\log \#A}$.

e. Fill the remaining entries of A_s and A_d with random strings.

f. For $1 \leq i \leq \#\mathbf{f}$ let $c_i \leftarrow \mathrm{SKE.Enc_{K_4}}(f_i)$.

g. Output $(\gamma, \mathbf{c})$, where $\gamma := (\mathrm{A_s, T_s, A_d, T_d})$ and $\mathbf{c} = (c_1, \dots, c_{\#\mathbf{f}})$.

– SrchToken(K_1, K_2, K_3, w): the client takes the key K and keyword w and outputs the corresponding search token as

$$\tau_s := \left(F_{K_1}(w), G_{K_2}(w), P_{K_3}(w)\right)$$

– Search$(\gamma, \mathbf{c}, \tau_s)$: the server takes the encrypted index, the search token and the ciphertexts of documents and return the search related ciphertexts in the following way

a. Parse τ_s as (τ_1, τ_2, τ_3) and return an empty list if τ_1 is not present in T_s.

b. Recover a pointer to the first node of the list by computing

$$\left(\alpha_1, \alpha_1^{'}\right) := \mathrm{T_s}[\tau_1] \oplus \tau_2$$

c. Look up $(v_1, r_1) = N_1 := \mathrm{A_s}[\alpha_1]$ and decrypt with τ_3 by computing

$$\left(\mathrm{id_1, addr_s(N_2)}\right) := v_1 \oplus H_1(\tau_3, r_1)$$

d. For $i \geq 2$, decrypt node N_i as above until $\alpha_{i+1} = \mathbf{0}.$

e. Let $\mathrm{I} = \{\mathrm{id}_1, \ldots, \mathrm{id}_m\}$ be the file identifiers revealed in the previous steps and output $\{c_i\}_{i\epsilon\ I}$, i.e., the encryptions of the files whose identifiers were revealed.

2.3. Computationally Efficient Dynamic SSE

In the [11] another dynamic searchable symmetric encryption scheme is presented providing searching in constant time in the number of unique keywords stored on the server. As usual each document M_i is characterized with a pair $(\mathrm{id_i}, \mathrm{W_i})$ where $\mathrm{ID_i}$ is the document's ID and $\mathrm{W_i} = \{\mathrm{w}_1, \mathrm{w}_2 \ldots\}$ is a set of keywords in document $\mathrm{M_i}$. For each document two keys are generated respectively $\mathrm{k_M}$ for encrypting a data M and $\mathrm{k_w}$ for encrypting a corresponding keyword. There is a searchable representation $\mathrm{S_w}$ for the keyword w based on $\mathrm{k_w}$. A trapdoor $\mathrm{T_w}$ corresponds to a pair $(\mathrm{w}, \mathrm{k_w})$. Search $(\mathrm{T_w}, \mathrm{S_w})$ outputs 1 if $\mathrm{w} \in \mathrm{W}$. Let $\mathrm{ID_i}$ be a document identifier (ID) and let $\mathrm{I_w}=\{\mathrm{ID_i} \mid \mathrm{w} \in \mathrm{W_i}\}$. The construction of the searchable representation for each keyword w is:

$$\mathrm{S_w} = (\mathrm{f}_{\mathrm{k_f}}(\mathrm{w}), \mathrm{m}(\mathrm{I_w}), \mathrm{R}(\mathrm{w}))$$

Where $\mathrm{f}_{\mathrm{k_f}}$ is a pseudo-random function that identifies the searchable representation of w, $\mathrm{k_f}$ is $\mathrm{k_w}$ or a part of it, m(.) is a masking function and R(w) is a function that keeps some information for unmasking $\mathrm{I_w}$. Each time the client wants to retrieve the encrypted data items containing keyword w a trapdoor $\mathrm{T_w}=(\mathrm{f}_{\mathrm{k_f}}(\mathrm{w}), \mathrm{R}'(\mathrm{w}))$ is sent to the server. The server first searches for $\mathrm{f}_{\mathrm{k_f}}(\mathrm{w})$ and if it occurs, the associated masked $\mathrm{m}(\mathrm{I_w})$ is unmasked using R(w) and $\mathrm{R}'(\mathrm{w})$. The server then sends the encrypted data items whose IDs occur in $\mathrm{I_w}$ to the client. The key idea of the second approach is to deploy a hash chain. A hash chain of length N,

$$\mathrm{H}^{\mathrm{N}}(\alpha) = \mathrm{H}(\mathrm{H}(\ldots.\mathrm{H}(\mathrm{a}) \ldots))$$

is constructed by applying repeatedly N times a hash function H(.) to an initial seed value a. Let $\mathrm{I_i}(\mathrm{w})$ be the set of IDs showing in which metadata item w occurs, which have been added to $\mathrm{S_w}$ in the i-th updating. The searchable representation of w after updating i times is:

$$S_{old}(w) = \left(f_{k_f}(w), E_{k_1(w)}(I_1(w)), H'(k_1(w)), \ldots\ldots E_{k_i(w)}(I_i(w)), H'(k_i(w))\right)$$

where $\mathrm{k_j(w)} = \mathrm{H}^{\mathrm{N-ctr}}(\mathrm{w}||\mathrm{k})$ and ctr is a counter which is incremented each time the storage is updated. The counter is stored in the client side. This construction encrypts each list $\mathrm{I_j(w)}$ with a unique secret key $\mathrm{k_j(w)}$ such that the server is able to compute the encryption key of previously added lists $\mathrm{k_{j-1}(w)}$ .... $\mathrm{k_1(w)}$ given the encryption key of $\mathrm{k_j(w)}$, by traversing the hash chain forward, but the server cannot compute $\mathrm{k_{j+1}(w)}$ since it cannot traverse the chain backward. Based on above explanation assuming that $\mathrm{S_{old}}$ (w) has already been updated j times, the next updating of $\mathrm{S_{old}}$ (w) is implemented as follows:

For each unique word w ∈ W

– construct $I_{j+1}(w) = \{ID_j \mid w \in W_{j,}\}$

– Increment the counter $ctr = ctr + 1$,

– Compute the secret key $k_{j+1}(w) = H^{N-ctr}(w||k)$,

– Encrypt $E_{k_{j+1}(w)}(I_{j+1}(w))$

– Send to the server the tuple $f_{k_f}(w), E_{k_{j+1}(w)}(I_{j+1}(w)), H\left(k_{j+1}(w)\right)$ which adds the received tuple to S_{old} (w).

Then the Trapdoor in fact is T_w=($f_{k_f}(w), H^{N-ctr}(w||k)$)= ($T_1$,T_2) and a search (S ,T_w) is implemented as follows: Given the trapdoor and the searchable representation S, search of T_1 and if T_1 occurs, compute $H(T_2)$, if $H(T_2) = H(k_i(w))$, decrypt I_i using T_2, otherwise keep computing $T_2 = h(T_2)$ until $H'(T_2) = H'(k_i(w))$. After $k_i(w)$ is computed, repeat the same procedure to compute the previously added list of document identifiers. Having decrypted $I_i(w), \ldots I_1(w)$ send the documents whose IDs occur in the lists.

2.4. Dynamic Searchable Encryption in Very-Large Databases:

In [10] a dynamic searchable encryption scheme is presented which efficiently searches the server-side database of tens of billions rows. A basic algorithm is constructed based on a generic dictionary construction and two other optimized versions are built on top of the basic algorithm. Here we cover the basic construction and refer the interested reader to the original paper for details of the algorithm's next extensions.

Following to the formalization of Curtmola et al. [5] the database $DB = (id_i, W_i)$ is a list of identifier /keyword-set pairs where $id_i \in \{0,1\}^k$ and $W_i \in \{0,1\}^*$. $W = \bigcup_{i=1}^{d} W_i$. And for a keyword $w \in W$ we write $DB(w)$ for $\{id_i : w \in ID_i\}$.

A dictionary implementation D consists of four algorithms $Create$, $Get, Insert, Remove$. Create takes a list of label-data pairs $\{(l_i, d_i)\}$, where each label is unique, and outputs the data structure γ. On input γ and a label l_i, $Get(\gamma, l_i)$ returns the data item with that label. On input γ and (l, d), $Insert(\gamma, (l, d))$, outputs an updated data structure that should contain the new pair (l, d).

Let $D = (Create, Get, Insert, Remove)$ be a dictionary implementation, F be a variable-input-length PRF, and $\Sigma = (Enc, Dec)$ be a symmetric-key encryption scheme. Below we detail the database setup and search algorithms.

SETUP THE DATABASE

– $K \leftarrow \{0,1\}^k$ and allocate the list L.

– For each $w \in W$

– $K_1 = Enc(K, 1||w)$ and $K_2 = Enc(K, 2||w)$

– Initialize the counter $c = 0$

– For each $id_i \in DB(w)$

$l \leftarrow F(K_1, c)$; $d \leftarrow Enc(K_2, id_i)$; $c++$

Add (l, d) to the list L in lexicographically order.

– Set $\gamma = Create(L)$

– Outputs the user key K and γ.

SEARCH THE DATABASE

Client: On input (K, w)

– $K_1 = Enc(K, 1||w)$ and $K_2 = Enc(K, 2||w)$

– Send (K_1, K_2) to the server

Server: On input (K_1, K_2)

– For $c = 0$ until Get returns NULL

$d \leftarrow Get(\gamma, F(K_1, c))$

$id \leftarrow Enc(K_2, d)$

– Return the list of revealed ids.

This scheme is made dynamic with a simple trick of using a dictionary γ^+ which is initially empty and to which a pair (l, d) is added with each keyword addition. Search for a keyword w is performed by the server by first searching γ as in the static case, then re-computing all labels corresponding to w in γ^+. The latter labels are computed using a w-specific key provided by the client and a running counter. Note that addition operations involving keyword w require the client to know the current value of the w-specific counter. For this, the scheme maintains a dictionary δ associating each keyword that was ever added via edit+ or add with its

current counter value. δ can be stored at the client or stored at the server and retrieved by the client for performing update operations. Next we define the method of allowing adding new documents to the index.

UPDATE THE DATABASE**:**

– Set $K^{+} \leftarrow F(K, 3)$

– For each $w \in W$

– $K_1^{+} = Enc(K^{+}, 1||w)$ and $K_2^{+} = Enc(K^{+}, 2||w)$

– $c \leftarrow Get(\ \delta, w)$ if c = NULL then $c = 0$.

– Set $l \leftarrow F(K_1^{+}, c);\ \ d \leftarrow Enc(K_2^{+}, id_i);\ \ c++$

– Insert (w, c) into δ and insert (l, d) into γ^{+}.

The search now implies the searching in both γ^{+} and γ dictionaries. The server gets $(K_1, K_2, K_1^{+}, K_2^{+})$ and performs the following actions.

– For $c = 0$ until Get returns NULL

$d \leftarrow Get\big(\gamma, F(K_1, c)\big)$

$id \leftarrow Enc(K_2, d)$

– For $c = 0$ until Get returns NULL

$d \leftarrow Get\big(\gamma^{+}, F(K_1^{+}, c)\big)$

$id \leftarrow Enc(K_2^{+}, d)$

This scheme can allow delete operations over the secure index only by keeping the revocation list of all deleted file ids and next check each search result against the revocation list.

3. Public-Key Keyword Search

In [12] the problem of searching in public-key encrypted data is studied. The main use case of this scheme is considered the email gateway which needs to check emails for specific words, such is «urgent» to route them accordingly. But the mail gateway should not learn anything more about the encrypted messages. Assuming Bob wants to send an encrypted email to Alice. To do so, Bob encrypts the email with usual public-key encryption scheme and then appends the searchable encryption of each keyword. To send a message M with keywords $w_1, w_2, \dots w_N$, Bob sends the following:

$$E_{A_{pub}}(M)||PEKS\big(A_{pub}, w_1\big)||\ \dots .||\ PEKS(A_{pub}, w_N)$$

Here A_{pub} is the public key of the recipient Alice. The point of this encryption is that Alice can give the email gateway a special trapdoor T_w so the gateway can

check whether the encrypted email contains the word w or not without learning anything more about the encrypted message or the certain word w.

According to [14], the Public-Key encryption scheme with keyword search (PEKS) is the defined as a tuple of following algorithms:

– $KeyGen(s)$: Takes a security parameter s and outputs the public/private key pair A_{pub} , A_{priv} .
– $PEKS(A_{pub}, W)$: For the public key A_{pub} and the word W produces the searchable encryptio of W.
– $Trapdoor(A_{priv}, W)$: Given Alice's private key A_{priv}, produces the trapdoor T_w for the word W.
– $Test(A_{pub}, S, T_w)$: Given the Alice's public key A_{pub}, a searchable encryption $S = PEKS(A_{pub}, W')$ and the trapdoor $Test(A_{pub}, S, T_w)$: $Trapdoor(A_{priv}, W)$, outputs 'Yes' , if 'W=W'', or 'No' otherwise:

The constuction of PEKS scheme is based on bilinear mappings defined as $e: G_1 \; x \; G_1 \rightarrow \; G_2$ between two groups G_1, G_2 of prime order p where

1. Given $g, h \in G_1$, there is an efficient way to compute the value $e(g, h) \in G_2$
2. For any integer x, y ∈ $[1, p]$ it is true that $e(g^x, h^y) = \; e(g, h)^{xy}$
3. If g is the generator of G_1, then the $e(g, g)$ is the generator of G_2

Two hash functions $H_1: \{0,1\}^* \rightarrow G_1$ and $H_2: G_2 \rightarrow \{0,1\}^{logp}$ are used as building blocks. The definition of scheme is described as follows

– $KeyGen(s)$: The input security parameter determines the size p of the groups G_1, G_2. The algorithm picks a random $\alpha \leftarrow Z_p$ and a generator g of G_1. Outputs the $A_{pub} \; = [g, h = g^{\alpha}]$ and $A_{priv} \; = \; \alpha$.
– $PEKS(A_{pub}, W)$: for the given word W first computes $t = \; e(H_1(W), h^r) \; \in G_2$ for a random $r \in Z_{\mathrm{p}}$ and then outputs the value $S = PEKS\big(A_{pub}, W\big) = [g^r, H_2(t)]$
– $Trapdoor(A_{priv}, W)$: Outputs $T_W = \; H_1(W)^{\alpha} \; \in G_1$
– $Test(A_{pub}, S, T_W)$: $S = \big[g^r, H_2\big(e(H_1(W), h^r)\big)\big] = [A, B]$. Checks if $H_2\big(e(T_W, \mathrm{A})\big) = B$.

The correctness of this scheme comes from the fact that $H_2\big(e(T_W, \mathrm{A})\big) = H_2\big(e(H_1(W)^{\alpha}, g^r)\big) = H_2\big(e(H_1(W) \, , g \,)\big)^{\alpha r}$ and $B = H_2\big(e(H_1(W), h^r)\big) =$ *H2eH1W,gαr=H2eH1W,gαr*

It is proven that this searchable encryption scheme is secure against a chosen keyword attack in the random oracle model assuming the BDH is intractable, mean-

ing that given the generator g of G_1 and three values $g^a, g^b, g^c \in G_1$, any polynomial time algorithms have a negligible advantages in computing the value $e(g,g)^{abc} \in G_2$.

4. Multi-Keyword Searchable Encryption Schemes

In the multi-key setting [13] there are set of users each having a set of documents and a server which stores the encrypted documents. Each user has access to a subset of documents. The user can create a new document and give access to that document to another users by sharing with them the decryption key.

The functionality goal is to allow a user to search a word over all the documents he can access, say n documents, even if those documents are encrypted under different keys. No te that the user has access to all the keys for these n documents, but the user should only give one search token to the server, instead of n tokens. A multi-key encryption scheme is a tuple of the following algorithms - (Setup, KeyGen, Delta, Token, Enc, Adjust, Match):

– $Setup()$: Takes the security parameter and returns the system wide params.
– $KeyGen(params)$: Takes the system parameters and returns a secret key such as the document encryption key or user's public key.
– $Delta(k_1, k_2)$: Takes two keys and returns the delta.
– $Token(k, w)$: Takes a keyword and a key and returns the search token tk.
– $Enc(k, w)$: Takes the keyword and key and outputs the encryption c of the word.
– $Adjust(tk, \Delta)$: Takes a search token tk $= Token(k_1, w)$ and a delta $\Delta = Delta(k_1, k_2)$and returns a search token tk'
– $Match(tk, c)$: Takes a search token and encrypted document c and outputs a bit b.

The key of user i is denoted with uk_i , and the key of document j with k_j. Consider that a user, say Alice, (with key uk_A) has n encrypted documents at the server, and each is encrypted under a key k_j for j = 1 . . . n. Alice wants to search for a word w over all the documents she has access to, so she uses ukA to compute a token for a word w. In order to allow the server to match the token against words encrypted with $k_1, \ldots, k_n$, Alice gives the server some public information called delta. Alice provides one delta per key k_j , denoted Δ_{uk_A, k_j}. The server can use Δ_{uk_A, k_j}. to convert a search token under key uk_A to a search token under k_j , a process we call adjust. In this way, the server can obtain tokens for word w under

k_j , . . . , k_j while only receiving one token from Alice, and then performing a traditional single-key search with the new tokens.

The constuction of the multi-key searchable encryption scheme is based on the asymmetric bilinear mappings defined as $e: G_1\ x\ G_2 \rightarrow\ G_T$ between groups G_1, G_2, G_T of prime order p where

1. Given $g, h \in G_1 x G_2$, there is an efficient way to compute the value $e(g, h) \in G_{\mathrm{T}}$
2. For any integer x, y $\in [1, p]$ it is true that $e(g^x, h^y) =\ e(g, h)^{xy}$
3. If g is the generator of G_1 and h is the generator of G_2, then the $e(g, h)$ is the generator of G_{T}

Let $H: \{0,1\}^* \rightarrow G_1$ and $H: G_T x G_T \rightarrow \{0,1\}^*$ be hash functions. The multi-key search scheme is defined as follows:

– $params \leftarrow Setup(1^k)$: Returns the system parameters $(p, G_1, G_2, G_{\mathrm{T}}, e, g_1, g_2, g_T)$.
– $k \leftarrow KeyGen(params)$ Returns $k \leftarrow Z_p$
– $\Delta \leftarrow Delta(k_1, k_2)$: Returns $\Delta = {g_2}^{k_2/k_1} \in G_2$
– $tk \leftarrow Token(k\ , w)$: Returns $tk = H(w)^k \in G_1$
– $c \leftarrow Enc(k\ , w)$: Select $r \leftarrow G_T$ and outputs $c = (r, H_2(r,\ e(H(w), g_2)^k))$
$tk' \leftarrow Adjust(tk\ , \Delta)$: returns $tk' =\ e(tk, \Delta)\ \in G_T$
– $b \leftarrow Match(tk'\ , c)$: Assuming $c = (r, h)$ outputs 1 if $H_2(r, tk')\ = h$, else 0.

The correctness of this scheme comes from the following observations: having $params \leftarrow Setup(1^k)$, $k_1 \leftarrow KeyGen(params)$, $k_2 \leftarrow KeyGen(params)$, $\Delta \leftarrow Delta(k_1, k_2)$, $tk \leftarrow Adjust(Token(k_1\ , w), \Delta)$ we can see that $tk = e\big(H(w)^{k_1}), {g_2}^{k_2/k_1}\big)\ = e(H(w), g_2)^{k_2}$ and $H_2(r, tk)\ =\ H_2\big(r, e(H(w), g_2)^{k_2}\big)$ meaning $Match\big(tk\ , Enc(k_1, w)\big) = True.$

5. Conclusion

Searchable encryption is one of the most important cryptographic disciplines under very active investigation nowadays. In this paper the recent searchable encryption algorithms are covered in details from three different settings exposing the most efficient constructions which have found different practical applications already.

REFERENCES

1. *Song D. X., Wagner D., and Perrig A.,* Practical techniques for searches on encrypted data: a. In Proceedings of the 2000 IEEE Symposium on Security and Privacy, Washington, DC, USA, 2000. IEEE Computer Society.

2. *Goldreich O. and Ostrovsky R.,* Software protection and simulation on oblivious RAMs. Journal of the ACM, 43(3): 431–473, 1996. 2

3. *Bellare M. and Rogaway P.,* Random oracles are practical: A paradigm for designing efficient protocols. In V. Ashby, editor, ACM CCS 93, pages 62–73, Fairfax, Virginia, USA, Nov. 3–5, 1993. ACM Press.

4. *Goh E.-J.,* Secure indexes. Cryptology ePrint Archive, Report 2003/216, 2003. http://eprint.iacr.org/2003/216/.

5. *Curtmola R., Garay J., Kamara S., Ostrovsky R.,* Searchable Symmetric Encryption: Improved Definitions and Efficient Constructions. In proc. ACM Conference on Computer and communication Security, pages 79– 88, 2006.

6. *Cash D., Jarecki S., Jutla C.S., Krawczyk H., Rosu M.-C. and Steiner M.,* Highly-scalable searchable symmetric encryption with support for boolean queries. In R. Canetti and J. A. Garay, editors, CRYPTO 2013, Part I, volume 8042 of LNCS. PP. 353–373.

7. *Kamara S., Papamanthou C. and Roeder T.,* Dynamic searchable symmetric encryption. In T. Yu, G. Danezis, and V. D. Gligor, editors, ACM CCS 12. PP. 965–976, Raleigh, NC, USA.

8. *Kamara S. and Papamanthou C.,* Parallel and dynamic searchable symmetric encryption. In A.-R. Sadeghi, editor, FC 2013, volume 7859 of LNCS. PP. 258–274, Okinawa, Japan, Apr. 1–5, 2013. Springer, Berlin, Germany.

9. *Liesdonk van P., Sedghi S., Doumen J., Hartel P., Jonker W.,* Computationally Efficient Symmetric Encryption. In Secure Data Management (SDM). PP. 87–100, 2010.

10. *Cash D., Jaeger J., Jarecki S., Jutla Ch., Krawczyk H., Rosu M.-C. and* Steiner M., Dynamic Searchable Encryption in Very-Large Databases: Data Structures and Implementation.

11. *Jarecki S., Jutla C., Krawczyk H., Rosu M.C. and Steiner M.,*Outsourced symmetric private information retrieval. In ACM CCS 13, Berlin, Germany, Nov. 4–8, 2013. ACM Press.

12. *Boneh D., Crescenzo di G., Ostrovsky R. and Persiano G.,* Public key encryption with keyword search. In Advances in Cryptology–EUROCRYPT ’04, volume 3027 of Lecture Notes in Computer Science, pages 506–522. Springer, 2004.

13. *Raluca A., Popa and Zeldovich N.,* Multi-Key Searchable Encryption. MIT CSAIL. https://people.csail.mit.edu/nickolai/papers/popa-multikey-eprint.pdf

ОБЗОР АЛГОРИТМОВ ШИФРОВАНИЯ, ДОПУСКАЮЩИХ ПОИСК

М. Овсепян

Российско-Армянский (Славянский) университет
hovsepyan.mihran@gmail.com

АННОТАЦИЯ

Шифрование, допускающее поиск данных, позволяет пользователю хранить свои данные в ненадежной среде, например, в общедоступном облачном хранилище, в зашифрованном виде, тем самым не ограничивая возможность поиска этих данных. При этом провайдер хранилища не имеет возможности читать данные, результаты поисковых запросов и даже сами запросы. Значимость такой функциональности растет в связи с ростом популярности облачных хранилищ – таких, как Dropbox и Google Drive, и, параллельно с этим, среди научного сообщества криптографии растет интерес к изучению этого вопроса. В данной статье содержится краткий обзор известных на данный день алгоритмов шифрования, допускающих поиск с практически применимыми характеристиками и поверхностный анализ сильных и слабых сторон этих схем.

Ключевые слова: криптография, облачное хранилище, безопасный индекс поиска.

ՈՐՈՆՎՈՂ ԳԱՂՏՆԱԳՐՄԱՆ ՄԵԹՈԴՆԵՐԻ ԱՄՓՈՓՈՒՄ

Մ. Հովսեփյան

Հայ-Ռուսական համալսարան
hovsepyan.mihran@gmail.com

ԱՄՓՈՓՈՒՄ

Որոնվող գաղտնագրման մեթոդները թույլ են տալիս օգտագործողին պահել իր տվյալները գաղտնագրված կերպով ոչ վստահելի միջավայրում, ինչպիսիք են ամպային բաց պահոցները, բայց միևնույն ժամանակ հնարավորություն են տալիս օգտագործողին կատարել անվտանգ որոնման գործողություն այդ գաղտնագրված տվյալներում: Ընդ որում ամպային պահոցների սերվերները չեն կարող կարդալ ոչ օգտագործողի տվյալները, ոչ նույնիսկ որոնվող հարցումները: Նման ֆունկցիոնալության կարևորությունը աճել է ամպային պահոցների (Dropbox, Google Drive) մաշտաբային կիրառմանը զուգահեռ և գիտության այս ճյուղը ստացել է մեծ ուշադրություն հետազոտողների կողմից: Այս հոդվածը ներկայացնում իրենից ներկայումս հայտնի կիրառական որոնվող գաղտնագրման մեթոդների կարճ ամփոփում, ինչպես նաև այդ մեթոդների թույլ և ուժեղ կողմերի հպանցիկ անալիզ:

УДК 62-50 Поступила 25.08.2015г.

STOCHASTIC DISCRETE EVENT SIMULATION MODEL FOR ESTIMATING PRODUCT DEVELOPMENT TIME

A. Mkrtchyan

Massachusetts Institute of Technology Cambridge, MA 02139, USA
armen@mit.edu

SUMMARY

Efficient development of products is critical to the success of many firms. The literature on product development often discusses the seemingly emergent nature of product development processes in various settings and organizations but does not adequately address the lack of planning tools within these organizations. This paper aims to fill the gap in tool support for the planning process by developing a discrete event simulation model to compute lead times of product development projects. The model explicitly captures various tasks, teams, and developers within a distributed product development setting. This work captures the iterative nature of the product development process and specifies the contextual relationships among tasks and among developers. The model was validated on a previously collected data set, as well as using data from an ongoing project with a small-sized software firm. The model has been encapsulated into a tool using an easy-to-use desktop and iPhone/iPad applications.

1. Introduction

The economic success of most firms depends on their ability to identify the needs of customers and introduce new products over time [1], [2]. The goal of product development (PD) process is to create these products. PD is defined as the set of activities beginning with the perception of a market opportunity and ending in the production, sale, and delivery of a product [3]. In today's environment of rapidly evolving customer preferences, speed and flexibility in developing new products is even more important [4].

Furthermore, geographically distributed PD is increasingly becoming more popular [5], [6]. Various researchers have noted the importance of developing products in distributed fashion (e.g., [7]) and with the emergence of reliable electronic-based communication media, firms are embracing distributed PD. Sproull and Kies-

ler [8] note that information technology (IT) reduces the dependence on traditional face-to-face communications and creates "networked organization." Hence, this paper investigates and presents a simulation model to enhance disctributed PD processes.

2. Model Description

This model employs a queuing-based discrete event simulation (DES) to estimate the lead time of product developmet projects. While in some circumstances it is possible to formulate PD process as a mathematical optimization problem, the complexity of real-world PD processes makes raw optimization computationally unviable. In DES systems utilized in this paper, time advances in discrete steps defined by elapsed time between events [9], hence, components of the system do not need to be scanned during times between events [10]. To build the DES model, the the main attributes affect PD processes were considered (discussed further by krtchyan and Srinivasan [11]). These attributes along with the requisite modeling variables are shown in Table I.

Table I: PD attributes and corresponding modeling variables.

PD Attribute	Variables
Task structure	– Task duration, T_d^i – Task type & task assignments, T_t^i
Team structure	– Number of teams, N_F – Number of developers per team, N_F^D – Team assignments, F_D
Developer flexibility	– Developer work hours, D_h – Developer work type, D_t
Rework	– Rework impact, $K_{i_q}^{j,rework}$. – Rework probability, $R_{i_q}^{j,rework}$
Task relatedness	– Task dependency, $T(i,j)$ – Task overlap, $O(i,j)$
Learning	– Productivity, P_D – Learning curve, L_D&L_T^{min}
Team coordination	– Coordination cost, ΔS_i^{Cross}

This DES model has four interconnected components as shown in Figure 1. First, the task model describes the initial tasks, including task types, duration, and

dependency between tasks. Next, the task rework model represents the rework structure. This includes rework probabilities, i.e., probability that one task causes rework for another task, and rework impact, which indicates rework duration. The third component is the developer model, which describes attributes pertaining to developers. These attributes are the number and type of developers, team assignments, productivity level, learning curve, developer priority, coordination cost, and work hours. The last component is the queue, which stores the tasks before they are serviced by the developers.

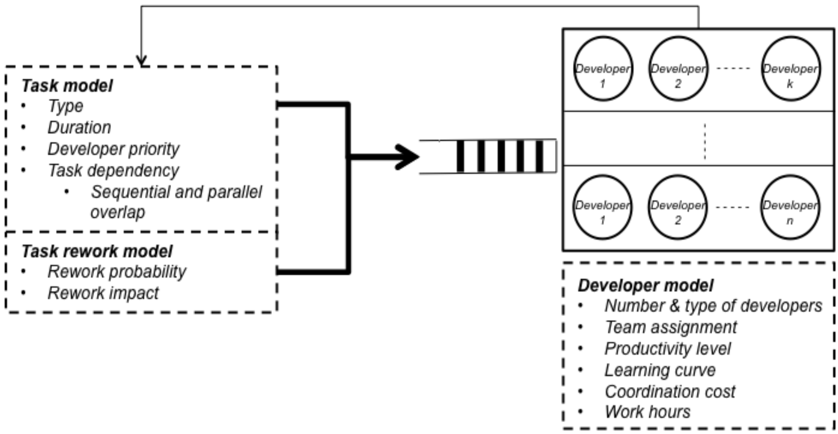


Figure1: High-level representation of the model.

Queuing-based DES models have been used extensively in a variety of domains, such as manufacturing, human supervisory control [12], hospitals [13], finance [14], and air traffic control [15]. DES models have also been successfully used to model different aspects of PD. For example, Adler et al. [16] utilized DES technique to model workflow management, while Browning and Eppinger [17] modeled the impact of process architecture on PD schedule. The variables identified above were incorporated into the DES model, which is described in the next sections.The model has been implemented both in the form of a desktop application running on Mac OS, as well as an iPad application. A sample screenshot of the MacOS application, called SimLink™is shown in Figure 2.

2.1. Model Structure

All of the model components are described below, along with input data required to run the model.

2.1.1. Tasks

There are two general categories of tasks. The first category represents the initial tasks that need to be completed. These are the tasks that the manager thinks need to be completed for a given project to succeed. The next category of events represents the tasks that are the result of the rework caused by incomplete information at the time of initial task breakdown and planning.

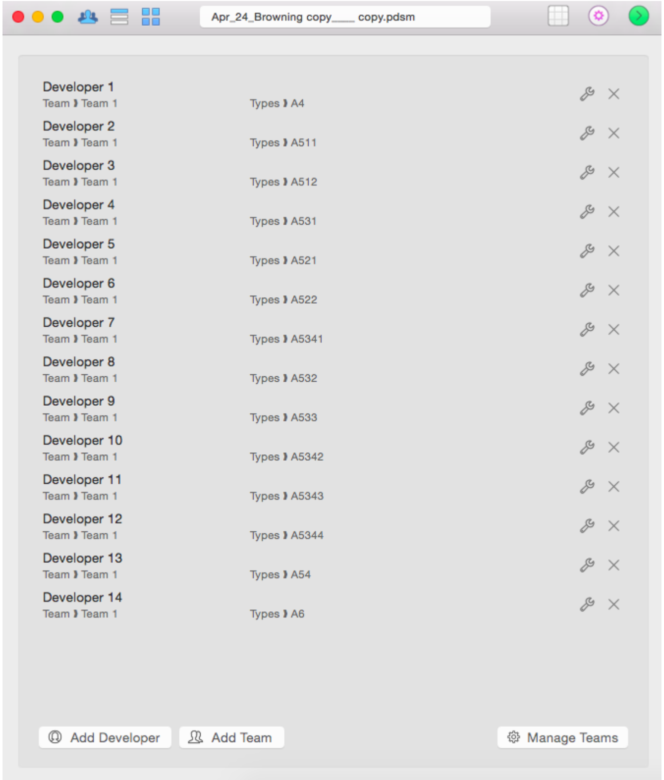


Figure 2: Sample screenshot of SimLink™ application.

Initial tasks

Initial tasks are pre-programmed in the system. Each task has its associated type and duration. When compiling a list of initial tasks, task dependencies are also specified. More specifically, the degree to which tasks can be processed in parallel and whether starting one task is dependent on finishing another are specified.

Rework generated tasks

The iteration of design/development tasks is critical to any PD process [18]. While product quality generally improves with each successive iteration, it also contributes significantly to the cost and completion time of a project [10]. Much of the rework is caused by changes in information and/or assumptions upon which they were initially executed. In this model, rework is taken into account using a design structure matrix (DSM) methodology to represent the probability that a rework is required, as well as the impact that the rework will have. These concepts are explained in more detail in the *Arrival Processes* and *Service Processes* sections.

2.1.2. Arrival Processes

Each task in a DES simulation has associated interrarival time, which describes the arrival of successive tasks. Arrivals can occur at random times or at scheduled times. When at random, the interarrival times are usually characterized by a probability distribution [9]. The arrivals can also be either independent or dependent. In this model, arrivals can be dependent based on when other tasks are serviced, which can be of the same or have a different task type. To model the dependency of tasks, information flow between tasks is specified through a DSM. Servicing a task can trigger other tasks based on the dependency DSM, which is discussed in the next sections. The newly triggered tasks subsequently enter the queue and are later serviced according to the queue discipline. Servicing a task can also cause other tasks to be unblocked, i.e., these tasks are allowed to leave the queue and be serviced. To model the blocking/unblocking [19], the flow of certain tasks in the queue is temporarily stopped until a condition is met (e.g., another task is serviced). The concept of dependency is extended in this model to account for tasks that may be unlocked after a certain percentage of a different task is finished, without waiting for the task to be fully completed (e.g., dependency of initial tasks, as described below).

Initial task interarrival times

Interarrival time of initial tasks is not explicitly modeled because availability of developers and dependency between tasks dictates when the initial tasks are serviced. To model dependency between tasks, information flow from task i to task j is specified through a dependency matrix D. $D(i,j)$ is zero when there is no information flow from task j to task i and the two tasks can be processed in parallel. $D(i,j)$ is one when there is information flow from task j to task i and the two tasks are either processed sequentially or there can be some overlap. To specify the level

of overlap between tasks j and i, an overlap matrix O is used. $O(i,j) = b$ implies that task i can be serviced before task j has been completed and the level of overlap is $T(j) \cdot b$, where $T(j)$ is the initial duration of task j.

Rework task interarrival times

Arrival of tasks associated with rework are modeled using a rework probability matrix $R_{i_q}^{j,rework}$, which indicates probabilities that task j causes rework for task i during qth iteration. More specifically, the value of $R_{i_q}^{j,rework}$ implies whether rework is caused for task i after qth iteration of task j, i.e., the value of $R_{i_q}^{j,rework}$ is the success probability of the Bernoulli distribution.

2.1.3. Service Processes

Similar to arrival processes, service processes can also be either constant or of random duration. Also, different tasks can have different service times (or probability distributions).

Initial task service times

To indicate the duration of initial tasks, the model allows product/project managers to input their pessimistic(p), likely (l), and optimistic (o) estimates. Specifically, a time duration matrix $T_d(j) = (p, l, o)$ uses latin hypercube sampling (LHS) method to generate expected values for the duration of task j. Latin hypercube sampling has been shown to have a better convergence rate compared to a more popular random sampling method [20], [21]. Since the estimate of initial service times of tasks is usually carried out by project managers based on their prior experience, this model captures the 10th percentile of the expected duration, the mode and the 90th percentile of the expected duration of a task as(p, l, o). It has been shown that the 10th and 90th percentiles of the expected duration are easier for humans to estimate than the 0th and 100th percentiles of the probability distribution function (PDF) of expected durations [10].

For each task j, its pessimistic, likely and optimistic duration estimates can be used to generate a PDF $f_{\Xi_j}(\xi)$. Specifically, given the 0^{th} (a) and 100^{th} (b) percentiles for task durations, as well as the mode (c), the PDF of a triangular distribution can be constructed. In fact, given $\begin{cases} a: -\infty < a < \infty \\ b: a < b \\ c: a \le c \le b \end{cases}$, one can write the PDF of a triangular distribution as follows.

$$f_{\Xi}(\xi) = \begin{cases} 0, for\ \xi < a\ or\ \xi > b \\ \dfrac{2(\xi - a)}{(b-a)(c-a)}, for\ a \le \xi \le c \\ \dfrac{2(b-\xi)}{(b-a)(b-c)}, for\ c < \xi \le b \end{cases}$$

To find a, c, b from known estimates of p, l, o, the following system of equations has been derived by Mkrtchyan [22].

$$\begin{cases} (1+z)^2 + (1+w)^2 = 10 \\ \dfrac{1}{l-o} z(z+1) = \dfrac{1}{o-l} w(1+w) \end{cases}$$

Using a publicly available library for numerical analysis and taking into account initial conditions (i.e., $z > 0, w > 0$), the values of z and w can be computed for any estimates of p, l, o. Next it can be shown that $a = o - \frac{1}{z}(l - o)$ and $b = p + \frac{1}{z}(p - l)$, while $c = l$[22].

Latin hypercube sampling (LHS)

To generate task duration samples when running the simulation model, the LHS method is used. LHS is a type of stratified sampling [21], [23] and it operates the following way when sampling data for K tasks. The range spaces for each component of ξ, i.e., $\xi_1, \xi_2, \ldots, \xi_K$ is split into M disjoint intervals (also known as stratas) on the basis of equal probability size, where M is the number of simulation iterations. In this case, each disjoint interval has a probability size of $1/M$. One value from each interval is selected randomly. Hence, M values are obtained for each component of ξ. Lastly, M values from ξ_1 are randomly combined with M values of $\xi_2, \xi_3, \ldots, \xi_K$ to form MK-tuples. The set of K-tuples is also known as Latin hypercube sampling. Therefore, for given M and K, there exist $(M!)^{K-1}$ interval combinations for a Latin hypercube sampling. Furthermore, compared to random Monte Carlo sampling method, LHS has significantly smaller sampling error of $O\left(1/_N\right)$[24], [25]., while the sampling error of Monte Carlo is $O\left(1/_{\sqrt{N}}\right)$, where N is the number of samples [26]. This means that the sampling error decreases quadratically faster in the case of LHS.

To accomplish the above-described process, an easy to implement method has been proposed, which uses inverse cumulative distribution function (CDF) [27], [28]. First, the interval $[0,1]$ is divided into M intervals. Next, the midpoint of each interval is determined. Afterward, using inverse CDF, values corresponding to task durations are extracted. Specifically, for a triangular distribution, random variates can be generated as follows knowing the value of $v \in [0,1]$.

$$\begin{cases} \xi = a + \sqrt{v(b-a)(l-a)}, & when\ 0 < v < \dfrac{l-a}{b-a} \\ \xi = b - \sqrt{(1-v)(b-a)(b-l)}, & when\ \dfrac{l-a}{b-a} \le v < 1 \end{cases}$$

M samples are generated this way for each component of ξ, which are used to specify task durations for K tasks during M simulation runs. Because this method chooses the midpoint for each of the M intervals, the method is referred to as median Latin hypercube sampling.

Rework service times

Rework service times are modeled as fractions of initial service times. For example, if task i requires a rework, its duration is modeled as a fraction of the initial task, i.e, $S_{i_q}^{rework} = k_{i_q}^{rework} \cdot S_i^{initial}$, where $S_i^{initial}$ is the initial duration of task i and $k_{i_q}^{rework}$ is the fraction of work that needs to be reworked for task i during qth iteration. The values of $k_{i_q}^{rework}$ are extracted from a three-dimensional matrix $K_{i_q}^{j,rework}$, which also includes information about which task (j) causes rework for task i during qth iteration.

2.1.4. Factors impacting service time

While service times are generally extracted from a probability distribution function, they can be further impacted by a variety of factors. Some of the major factors impacting service times are the learning curve, individual developer performance variations, and coordination cost.

The learning curve measures a characteristic of a task when it repeats. This model assumes that the duration of task i decreases by L_i percentage every time task i (or a portion of it) repeats until reaches some L_i^{max}, which is the maximum possible gain from repeating task i.

Another factor that can impact the service time is the individual differences between developers. More specifically, varying levels of developer experience, knowledge, and commitment levels can impact their performance, which in turn

impacts task service times. To take this into account, the model allows the project manager to account for performance differences by specifying the performance level for each developer. Specifically, the model assumes a default performance level of 1. If a developer is assumed to have better than average performance, the manager can increase the performance level by increasing the number, e.g., a developer with a performance level of 1.5implies that tasks will be completed 50% faster compared to a developer with a performance level of 1.

Lastly, the model also takes into account coordination cost between developers. The coordination cost refers to the amount of extra time that one developer needs to get accustomed to a task that another developer had been working on. The coordination cost is assumed to be zero for the same person, i.e., when a developer continues his own work there is no extra coordination cost that should be taken into account. Furthermore, the coordination cost is higher for developers that work in different teams and/or in different sites and is generally lower for developers working side by side in the same team/site. The model accounts for the coordination cost by adding ΔS_i^{Cross} to the service time of task i when a developer from a different team previously worked on the same task. In this case, ΔS_i^{Cross} is known as the cross-site coordination cost for task i.

2.1.5. Team Structure

To account for various team structures and their effect on product development process, the model takes into account the following factors.

Developer types

Developer types can be entered in the model, which is used to allocate tasks from the queue. Since all the tasks have their appropriate types, it is required that all task types have at least one matching developer with the same type. Otherwise, a task will never be serviced, if a developer with the same type does not exist.

Number of developers & priority levels

Developers can be of different type of they can have the same type. If developers have the same type, their priorities can be different. More specifically, two developers can be structural engineers but if one of them has a higher priority, a structural engineering task will be allocated to the developer with the higher priority, if he/she is available.

Work hours

Different teams can have different work hours in the model. This is especially useful when modeling a distributed product development process that includes teams in multiple times zones/countries. Knowing work hours of different teams helps take into account task distributions between developers working in different time zones, as well as modeling coordination cost between developers.

2.1.6. Queue

Queue serves as a temporary holding place for tasks before developers service them. However, besides serving just as a storage for the tasks, the queue has a discipline, which describes the logical ordering of tasks in a queue. It determines which task will be serviced first when a developer (server) becomes available [9]. There are numerous queue disciplines (e.g., first-in-first-out, service in random order, shortest processing time first) that can be implemented in a DES model. In this model, service according to priority queue discipline is implemented, which allocates highest priority task to the first available developer of the same task type (with the highest priority).

2.1.7. Servers/Developers

The developers in this model represent the servers of a queuing based DES model. Since developers can work simultaneously, they can be viewed as N parallel servers, where$N$ is the number of developers/servers. Nonetheless, it should be noted that these servers have varying characteristics (e.g., performance ratings, types of tasks they can service, work hours) and they do not always have to work in parallel. For example, if two developers work on a project that consists of two dependent tasks of different types, then one developer has to wait until the other developer finishes the task. This occurs despite the fact that both developers/servers, in theory, can work in parallel. Another important characteristic that describes the servers are work hours. In this model, the servers only become available during work hours and are inactive otherwise.

3. Model Outputs

The model is able to capture a variety of metrics that are useful for analyzing product development process. Specifically, being a queuing-based DES model, it is easy to capture long-run measures of performance of this queuing system. This me-

trics are the following: total and average time spent in the queue, total and time-average number of tasks in the queue, utilization of each server. Besides these DES metrics, the model also captures the total time it takes developers to complete all the tasks.

3.1. DES-based metrics

The main steady-state DES-based metrics are (a) the total and average time tasks spend in the queue, (b) total and time-average number of tasks in the queue, (c) average utilization and utilization of each server/developer. In this model, from the steady-state measures mentioned above, utilization can be used as a measure of developer workload. It is calculated as the ratio of the time the developer is busy servicing tasks divided by the total duration of the simulation. For a single server queuing system, the long run server/developer utilization (ρ) is equal to the average event arrival rate (λ) divided by the average service rate (μ).

$$\rho = \frac{\lambda}{\mu}$$

For the queuing system to be stable, the arrival rate must be less than the service rate, i.e., $\lambda < \mu$. If the arrival rate is greater than the service rate, then $\rho = 1$. In real-world situations, this can happen when developers have more tasks than they can service. One way to alleviate the saturation of servers is to increase the number of developers. Since real-world PD projects usually have multiple developers working in parallel, the notion of average developer utilization can also be computed, i.e., the simple arithmetic mean of the developers' utilization.

In this model, average task wait time in the queue is also calculated. To find the average time tasks spend in the queue, we define $W_1^Q, W_2^Q, \dots, W_N^Q$ to be the time each task spends in the queue, where N is the number of arrivals during $[0, \tau]$. Hence, the average time spent in the queue per event will be:

$$\widehat{w}_Q = \frac{1}{N}\sum_{i=1}^{N} W_i^Q$$

As $N \to \infty, \widehat{w}_Q \to w_Q$, where w_Q is the steady-state time spent in the queue. For stable queuing systems, w_Q must be bounded, otherwise wait times will grow indefinitely.

Similarly, let τ_{u} denote the total time during $[0, \tau]$ in which the queue contained exactly u tasks. The time weighted average number of tasks in the queue is defined by:

$$\hat{L}_Q = \frac{1}{\tau}\sum_{u=1}^{\infty} i\tau_u^Q = \frac{1}{\tau}\int_0^{\tau} L_Q(t)dt$$

As $\tau \to \infty, \hat{L}_Q \to L_Q$, where L_Q is the long-run time-average number of tasks waiting in queue.

3.2. PD specific metrics

Besides these DES metrics, the model also captures the total time it takes developers to complete all the tasks. Note that due to the ability to process tasks in parallel; the total time to service the tasks is generally less than the sum of times each developer spends on servicing tasks allocated to him/her. Assuming τ_L^g indicates the time when the last task in the queue was serviced by a developer during gth run of the model, the following metrics are calculated:

– Minimum project completion time: $\tau_{min} = \min_g(\tau_L^g)$.
– Maximum project completion time: $\tau_{max} = \max_g(\tau_L^g)$.
– Average project completion time: $\tau_{av} = \frac{1}{G}\sum_{g=1}^{G}\tau_L^g$, where G is the total number of simulation runs.
– Standard deviation of project completion time: $\tau_{SD} = \sqrt{\frac{1}{G}\sum_{g=1}^{G}(\tau_L^g - \tau_{av})^2}$.
– Median value of project completion time:

$$\tau_{med} = value of \tau_{med} for which P(X \le \tau_{med}) = P(X \ge \tau_{med}) = \frac{1}{2},$$

where is$X$ is the random variable representing project completion time after each simulation run.For each simulation run g, τ_D^g is defined as the time spent by developer D on servicing tasks.

The model is able to capture a variety of metrics that are useful for analyzing the PD process. Specifically, being a queuing-based DES model, it is easy to capture long-run measures of performance of this queuing system. Moreover, the model also captures PD specific metrics.

4. Results

4.1. Unmanned vehicle development project

To validate the model, the SimLink™ application was tested on an existing (also referred to as historical) data set gathered from industry. Specifically, Browning[18] collected information from Boeing on task durations, learning factors, task relatedness, and rework.

Table 2 shows task duration estimates. It should be noted that in this case the duration estimates represent the $0th$ percentile, mode, and the $100th$ percentile of a triangular distribution, rather than the $10th$ and the $90th$ percentiles used in the SimLink™ model. Also, in Browning's work resource constraints are not accounted for, hence the tasks can be processed in parallel to the extent allowed by task relatedness. To incorporate this into the SimLink™ model, each task type has its designated developer that can process the task once it arrives in the queue.

Table 2: Task duration estimates for a UV project.

	Durations (days)			
Tasks	o	l	p	**Learning**
A4	1.9	2	3	0.35
A511	4.75	5	8.75	0.2
A512	2.66	2.8	4.2	0.6
A531	9	10	12.5	0.33
A521	14.3	15	26.3	0.4
A522	9	10	11	1
A5341	7.2	8	10	0.35
A532	4.75	5	8.75	1
A533	18	20	22	0.25
A5342	9.5	10	17.5	0.5
A5343	14.3	15	26.3	0.75
A5344	13.5	15	18.8	0.3
A54	30	32.5	36	0.28
A6	4.5	5	6.25	0.7

Next, task relatedness is specified through a DSM, which is shown in Table 3. The values of the DSM indicate information flow between corresponding tasks. The super-diagonal elements indicate precedence relationships, while the sub-diagonal elements indicate feedback relationships between tasks.

It should be noted that the overlap matrix $O(i,j)$ is zero for this project, since Browning did not collect information and it was assumed that partial task overlap was not applicable.

Similarly, both rework probability and impact matrices[18] were captured and included in the model. After running the model $N = 200$ times, the SimLink™ results for project completion time are the following:

$$\begin{cases} Average = 151.6\ days \\ Standard\ deviation\ (SD) = 15.9\ days \end{cases}$$

Table 3: Task relatedness matrix for UV project.

	1	2	3	4	5	6	7	8	9	10	11	12	13	14
1														
2	1								1					
3		1		1										
4	1		1											
5	1		1			1		1				1	1	
6	1				1									
7	1					1								
8						1						1		
9	1		1	1				1						
10				1		1	1	1			1			
11						1	1	1		1				
12	1					1	1			1	1			
13	1				1							1		
14	1	1	1	1	1	1	1	1	1	1	1	1	1	

The results are compared to Cho and Eppinger's [10] modeling work, which also utilizes DES simulation and uses Browning's UV data set. Specifically, the results from the previous (simpler) model were:

$$\begin{cases} Average = 146.8 \\ Standard\ deviation = 17.0 \end{cases}$$

The results show that the SimLink™ model in a simple form is able to replicate the results using a historical data set. Also, the PDF of project completion time is positively (right) skewed (Figure 3), which agrees with the results from Cho and Eppinger [10] and Browning [18].

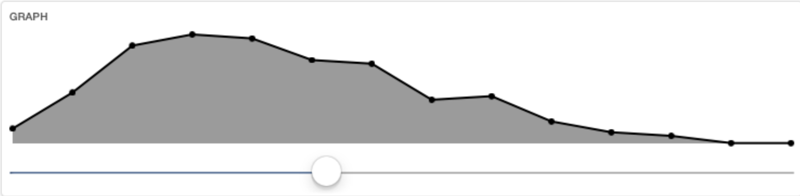


Figure 3: The PDF of project completion time for UV project.

4.2. Mobile software development project

This project is concerned with revising an existing system to better support the growing demands of a small firm's corporate clients. At the time this firm used Pivotal Tracker, an agile project management tool. Some of the information about the project, such as the tasks and developers working on the project were extracted from Pivotal Tracker. The rest of the required information was collected from a team leader responsible for this project. **Ошибка! Источник ссылки не найден.** shows the tasks, their types, and duration estimates.

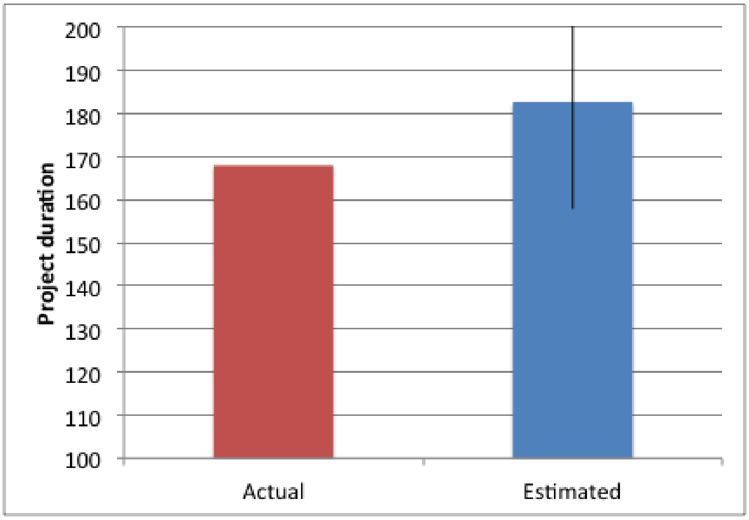
Table 4: Developer characteristics for mobile soft. dev. project.

Deve-loper	Skills / task types	Pruducti-vity level	Learning curve	Region	Typical work hours
1	– Games back end – API integration	0.75	0.2	Armenia	9:00–20:00
2	– Customer API	1	0.1	California, USA	20:00–06:00
3	– API integration	1.3	0.1	California, USA	20:00–06:00
4	– Customer API – Advertisement back end – Games back end	1.4	0.2	Armenia	9:00–20:00

Four developers were involved in this project. To be more precise, two developers from California and two developers from Armenia worked on the project as a small geographically distributive software development team. The developers had different performance levels, learning curves, and various skills. Table 4 summarizes the main characteristics of developers, as well as typical work hours (indicated in Armenian time).

The remaining modeling parameters, such as task relatedness and rework matrices are shown in Mkrtchyan's work [22].
Before the project was complete, the simulation model was utilized to predict project completion time. Moreover, the timeline was analyzed to identify opportunities to increase team efficiency and compare them with real world observations. The results (Figure 4) show that model predictions for average project completion time was 182.7 with SD of 25.84, while actual project completion time was 168 hours. While actual results were within one SD of model predictions, lower actual project completion time was mainly due to increased concurrency than what was accounted in the model. Specifically, the overlap matrix, $O(i,j)$ was specified as having all zero elements (i.e., tasks that exchange information do not overlap), however, in the real world developers collaborated more than anticipated by the PD manager and some overlap between tasks was present. Still the estimated results were within one SD of the actual project completion time.

Figure 4: Comparing actual vs. predicted project completion times.



5. Discussion

In general, three key areas for model applications were identified during the development and validation process, which are discussed below.

– *Setting schedule target* is one of the main potential applications of the model. PD managers can evaluate different scenarios and choose one that has an acceptable risk level of schedule overrun. The SimLink™ model can be used to

analyze only those PD scenarios that are feasible. For example, knowing how many developers can potentially work on a project, the SimLink™ should be utilized to evaluate only the scenarios that involve these developers, rather than some theoretical schedule tradespace that includes imaginary developers.

– *Process improvement* is another potential application area. Specifically, the model can help identify critical tasks that have significant impact on the PD process. By paying more attention to these tasks and allocating sufficient resources, the PD manager can improve overall PD process.

– *Research oriented evaluation* has also been identified as a potential application area of the model. There are PD strategies that have been consistently discussed in the PD literature but are usually hard to implement. The model can help quantitatively evaluate the impact of such strategies. For example, preemptive iteration has been suggested to shorten PD completion time by doing more iterations in the beginning of PD projects. While it is often risky to attempt such a strategy, the model can provide a quantitative measure of the expected reduction of PD project completion time.

Another example of a PD strategy that is hard to implement but easy to test with the model is follow-the-sun PD process implementation. Such a strategy requires setting up development activities in different time zones so that PD work can be passed from one time zone to another and effectively continues for 24 hours.

6. Conclusions

In this paper a novel method for predicting distributed PD lead time was presented. The above-described method uses stochastic DES approach to generate probability density functions of lead times. While lead times are one of the main parameters that interest project and product managers, the model can be used to analyze bottlenecks and optimize resources, conduct replanning activities, and improve PD processes in place. Given the level of granularity of the model, to small teams conducting PD activities. The modeling approach is valid for any PD process that has one or more teams of developers, including both hardware and software (and mixed) development projects. Furthermore, user-friendly Mac OS® and iOS® applications were developed to allow rapid dissemination of the model and make it easy for product/project managers to learn and use the model effectively.

REFERENCES

1. *Dougherty D. and Hardy C.*, Sustained product innovation in large, mature organizations: Overcoming innovation-to-organization problems, Acad. Manage. J., vol. 39, no. 5. PP. 1120–1153, 1996.
2. *Penrose E.T.*, The Theory of the Growth of the Firm. Oxford University Press, 1995.
3. *Ulrich K.T. and Eppinger S.D.*, Product design and development, 5th ed. New York: McGraw-Hill, 2012.
4. T*akeuchi H. and Nonaka I.*, The new new product development game, Harv. Bus. Rev., vol. 64, no. 1. PP. 137–146, 1986.
5. *Granstrand O., Håkanson L. and Sjölander S.,* Internationalization of R&D–a survey of some recent research, Res. Policy, vol. 22, no. 5. PP. 413–430, 1993.
6. *Griffin A.*, PDMA research on new product development practices: updating trends and benchmarking best practices, J. Prod. Innov. Manag., vol. 14, no. 6. PP. 429–458, 1997.
7. *Ghoshal S. and Bartlett C.A.*, The multinational corporation as an interorganizational network," Acad. Manage. Rev., vol. 15, no. 4. PP. 603–626, 1990.
8. *Sproull L. and Kiesler S.*, Connections: New ways of working in the networked organization. MIT press, 1992.
9. *Banks J., Carson J. II, Nelson B. and Nicol D.*, Discrete Event System Simulation, 5th ed. Harlow England: Pearson Education, 2014.
10. *Cho S.-H. and Eppinger S.*, Product development process modeling using advanced simulation," 2001.
11. *Mkrtchyan A. and Jayakanth S.*, Enhancing product development capabilities of SMEs: a study in a less-developed country, presented at the 21-st International Product Development Conference, 2014.
12. *Mkrtchyan A.A.*, Modeling operator performance in low task load supervisory domains, Massachusetts Institute of Technology, 2011.
13. *Ceglowski R., Churilov L. and Wasserthiel J.*, Combining data mining and discrete event simulation for a value-added view of a hospital emergency department, J. Oper. Res. Soc., vol. 58, no. 2. PP. 246–254, 2006.
14. *Boyle P.P.*, Options: A monte carlo approach, J. Financ Econ., vol. 4, no. 3, PP. 323–338, 1977.
15. S. Kapralov and V. Dyankova, "Modeling a system with discrete events," Comput. Model. Simul. EMS 2012 Sixth UKSimAMSS Eur. Symp. On, pp. 167–172, Nov. 2012.
16. *Adler P.S., Mandelbaum A., Nguyen V. and Schwerer E.*, From project to process management: an empirically-based framework for analyzing product development time, Manag. Sci., vol. 41, no. 3, pp. 458–484, 1995.
17. *Browning T. and Eppinger S.*, Modeling impacts of process architecture on cost and schedule risk in product development, Eng. Manag. IEEE Trans. On, vol. 49, no. 4. PP. 428–442, 2002.

18. *Browning T.,* Modeling and analyzing cost, schedule, and performance in complex system product development, MIT, Cambridge, MA, 1999.

19. *Balsamo S., Personé V. de Nitto and Onvural R.,* Analysis of queueing networkswith blocking, vol. 31. Springer, 2001.

20. *Chrisman L.,* Latin hypercube vs. Monte Carlo sampling, Lumina Blog, 23-Jul-2014. [Online]. Available: http: //blog.lumina.com/2014/latin-hypercube-vs-monte-carlo-sampling/. [Accessed: 15-Nov-2014].

21. *McKay M.D., Beckman R. J. and Conover W.J.,* Comparison of three methods for selecting values of input variables in the analysis of output from a computer code," Technometrics, vol. 21, no. 2. PP. 239–245, 1979.

22. *Mkrtchyan A.A.,* Modeling distributed product development processes in small and medium enterprises, Massachusetts Institute of Technology, Cambridge, MA, 2015.

23. *Keramat M. and Kielbasa R.,* Latin hypercube sampling Monte Carlo estimation of average quality index for integrated circuits, in Analog Design Issues in Digital VLSI Circuits and Systems, Springer, 1997. PP. 131–142.

24. *Aistleitner C., Hofer M. and Tichy R.,* A central limit theorem for Latin hypercube sampling with dependence and application to exotic basket option pricing, Int. J. Theor. Appl. Finance, vol. 15, no. 7, 2012.

25. *Loh W.-L.,* On the Convergence Rate to Normality of Latin Hypercube Sampling U-Statistics, Purdue University, Technical Report 95–2, 1995.

26. *Koehler E., Brown E. and Haneuse S.J.-P.,* On the assessment of Monte Carlo error in simulation-based statistical analyses, Am. Stat., vol. 63, no. 2. PP. 155–162, 2009.

27. *Keramat M. and Kielbasa R.,* Modified Latin hypercube sampling Monte Carlo (MLHSMC) estimation for average quality index, Analog Integr. Circuits Signal Process., vol. 19, no. 1. PP. 87–98, 1999.

28. *Wyss G.D. and Jorgensen K.H.,* A user's guide to LHS: Sandia's Latin hypercube sampling software, SAND98-0210 Sandia Natl. Lab. Albuq. NM, 1998.

СТОХАСТИЧЕСКАЯ МОДЕЛЬ ДЛЯ ОЦЕНКИ ВРЕМЕНИ РАЗВИТИЯ ПРОДУКТОВ НА ОСНОВЕ ДИСКРЕТНО-СОБЫТИЙНОЙ СИМУЛЯЦИИ

А.А. Мкртчян

АННОТАЦИЯ

Эффективное использование продуктов имеет особую важность для успешного развития многих фирм. В литературе по продвижению продуктов часто обсуждается, казалось бы, природа процессов их развития в различных условиях и организациях, но не говорится адекватно об отсутствии инструментов планирования в самих организациях.

Целью данной статьи является заполнение пробелов при поддержке инструментов с целью планирования и для создания моде-

лей дискретных событий, а также для посчета времени для выполнения новых проектов. Модель в рамках распределенной установки разработки продукта тщательно фиксирует разные задачи, группы и разработчиков. Эта процедура фиксирует повторяющуюся природу процессов использования продуктов и указывает на контекстуальные соотношения между задачами и разработчиками. Модель с помощью использования данных из текущих проектов была подтверждена на ранее собранном наборе данных, так же, как и у взятых малых фирм программного обеспечения. Модель преобразована в инструмент, который легко можно использовать как на мониторе компьютера, так и на iPhone/iPod-приложениях.

ՍՏՈԽԱՍՏԻԿ ԴԻՍԿՐԵՏ ԻՐԱԴԱՐՁՈՒԹՅՈՒՆՆԵՐԻ ՍԻՄՈՒԼՅԱՑԻՈՆ ՄՈԴԵԼ ԱՐՏԱԴՐԱՆՔԻ ՍՏԵՂԾՄԱՆ ԺԱՄԱՆԱԿԸ ԳՆԱՀԱՏԵԼՈՒ ՀԱՄԱՐ

Ա. Մկրտչյան

ԱՄՓՈՓՈՒՄ

Արտադրանքի արդյունավետ կազմակերպումը կարևոր է բազմաթիվ ընկերությունների համար: Արտադրանքի ստեղծման մասին գրականությունը հաճախ քննարկում է արտադրանքի առերևույթ զարգացման գործընթացների բնույթը տարբեր պայմաններում և կազմակերպություններում, սակայն լիարժեք չի անդրադառնում պլանավորման գործիքների բացակայությանը այդ կազմակերպություններում: Այս հոդվածը նպատակ ունի լրացնել այդ բացը առաջարկելով պլանավորման գործընթացի ստեղծման աջակցությանը նպաստող գործիք՝ օգտագործելով դիսկրետ իրադարձությունների սիմուլյացիոն մոդել: Մոդելը հստակորեն ներառում է տարբեր առաջադրանքներ, թիմեր և աշխատակիցներ՝ տեղաբաշխված արտադրանքի ստեղծման շրջանակներում: Այս աշխատանքը հաշվի է առնում արտադրանքի ստեղծման գործընթացի կրկնվող բնույթը և ամրագրում է բովանդակային հարաբերությունները առաջադրանքների և ծրագրավորողների միջև: Մոդելվավերացվել է նախկինում հավաքագրված տվյալների բազայի հիման վրա, ինչպես նաև օգտագործելով տվյալներ փոքր ծրագրային ընկերության արդիական ծրագրից: Մոդելի հիման վրա ստեղծվել է հեշտ օգտագործվող գործիք, որը կարելի է գործարկել համակարգչով եւ iPhone / iPad սարքավորումներով:

УДК 004.4 Поступила 12.11.2015г.

ИССЛЕДОВАНИЕ ВОЗМОЖНОСТИ ПРИМЕНЕНИЯ МАТЕМАТИЧЕСКИХ МЕТОДОВ АНАЛИЗА МУЗЫКАЛЬНЫХ ПРОИЗВЕДЕНИЙ

Л.А. Азнаурян

e-почта: lusine.aznauryan8@gmail.com

АННОТАЦИЯ

Данная статья посвящена исследованию математических методов раскрытия объективных механизмов описания и анализа музыкальных произведений, гармонии, исполнительства, «технологии» творчества и практической реализации художественно-эстетических замыслов. В работе выполнен исторический обзор математических методов, открытых еще в IV до н.э. Пифагором и Архитом, в средних веках – Иоганном Кеплером и немецким органистом Андреасом Веркмайстером. В качестве математических способов анализа звуков и звуковых событий предлагаются методы теории нечетких множеств, которые послужат основой для разработки алгоритмов гармонического анализа посредством логики нечеткого вывода.

Ключевые слова: звуки и звуковые события, звуковысотные соотношения, гармонические интервалы, темперация, Fuzzy Logic.

В статье раскрываются математические методы анализа музыкальных произведений, которые, как и другие методы, широко используются в современной педагогике, могут применяться и в учебном процессе, направленном на формирование мышления музыканта и, в частности, способствовать более эффективному музыкально-теоретическому обучению теории музыки, гармонии, анализа музыкальных произведений, полифонии и сольфеджио. Специфика музыки, музыкально-теоретических дисциплин, а также психологические особенности учащихся в музыкальных заведениях, дают предпосылки к обнаружению определенных форм алгоритмов, применимых только в данной сфере.

После обнаружения любопытной работы П.А. Черватюка, который применил систему алгоритмов в своем учебнике гармонии в 2-х частях (П.А. Черватюк Научно-методические основы преподавания гармони системой алго-

ритма; М.: изд. «Советский композитор, 1990г.), были изучены математическе методы построения алгоритмов и имитационных моделей, а также уже существующие приемы анализа музыкальных произведений.

Использование математических методов в музыкальной педагогике, продолжающее традиции музыкального образования, является естественным способом оптимизации учебного процесса в системе музыкального обучения. Далее, в статье приводятся доводы в пользу целесообразности разработки новых, более совершенных математических методов анализа музыкальных звуков, звуковых событий с целью внедрения их в процесс обучения музыкально-теоретическим дисциплинам (теории музыки, гармонии, анализу музыкальных произведений, полифонии и сольфеджио) взамен существующих. Конечной целью исследования является создание компьютерных программ на основе алгоритмов, которые позволят облегчить творческий труд музыканта.

В рамках данной статьи ниже приводятся некоторые изыскания, позволяющие положить их в основу программы, автоматизирующей процесс гармонического анализа музыкального произведения.

Прежде, чем перейти к разбору понятий нечеткого множества, которые вполне согласуются с интуитивными представлениями об окружающем мире и, в частности о музыке, следует определить связь математики и музыки, которая обусловлена как исторически, так и внутренне.

Еще в IV веке до н.э. великий Пифагор, создавший так называемый пифагорейский союз, и его соратник Архит – крупный теоретик в области пифагорейской музыки создали музыкальную систему [1], в основе которой были два закона, которые носят имена этих двух замечательных ученых. Эти законы постулировали:

Две звучащие струны определяют консонанс, если их длины относятся как целые числа, образующие треугольное число 10=1+2+3+4, т.е. как 1:2, 2:3, 3:4. Причем, чем меньше число n в отношении n/(n+1), где n=1, 2, 3, тем созвучнее получающийся интервал.

Частота колебания w звучащей струны обратно пропорциональна ее длине *l*:

w= a/*l*, (*l* – длина струны, a – коэффициент, характеризующий физические свойства струны).

Для понимания описанных далее связей музыкального анализа с теорией нечеткой логики и звуковысотной конструкции потребуются несколько

понятий теории музыки, в частности, гамм, интервалов между тонами и лада.

Гаммой, или звукорядом, называется последовательность звуков, расположенных в восходящем или нисходящем порядке от основного тона (звука). Интервалом между тонами называется порядковый номер ступени верхнего тона относительно нижнего в данном звукоряде, а интервальными коэффициентами двух тонов – отношение частоты колебаний верхнего тона к частоте нижнего: w_2/w_1. Математическое выражение системы звуковысотных соотношений – лада называется музыкальным строем. Лад – это приятная для слуха взаимосвязь музыкальных звуков, определяемая зависимостью неустойчивых звуков от устойчивых и имеющая определенный характер звучания. Основой музыкальной шкалы-гаммы пифагорейцев был интервал – октава. Она является консонансом, повторяющим верхний звук. Для построения музыкальной гаммы пифагорейцам требовалось разделить октаву на красиво звучащие части. Так как они верили в совершенные пропорции, то связали устройство гаммы со средними величинами: арифметическими, геометрическими, гармоническими. Что касается музыки средних веков, то уже тогда были определены интервальные коэффициенты и соответствующие им интервалы. Они были названы совершенными консонансами и получили следующие названия: октава (w_2/w_1= 2/1, l_2/l_1=1/2); квинта (w_2/w_1=3/2, l_2/l_1= 2/3); кварта (w_2/w_1=4/3, l_2/l_1 = 3/4). Звуки в музыкальной гамме связаны между собой определенными зависимостями. Одни из них являются неустойчивыми и тяготеют к другим, устойчивым. В каждой гамме есть наиболее устойчивый, основной тон. Он называется «тоникой», и с него начинается данная музыкальная система.

Идея совершенства окружающего мира владела умами ученых и в последующие эпохи. В первой половине XVII века Иоганн Кеплер (немецкий математик, астроном, механик, оптик, первооткрыватель законов движения планет Солнечной системы) установил семь основных гармонических интервалов: октаву – 2/1, большую сексту – 5/3, малую сексту – 8/5, чистую квинту – 3/2, чистую кварту – 4/3, большую терцию – 5/4 и малую терцию – 6/5. С помощью этих интервалов он вывел весь звукоряд как мажорного, так и минорного наклонения.

XVIII век открыл новые страницы в истории музыки. Основное открытие в этой области сделал немецкий органист А. Веркмайстер. Он осуществил гениальное решение, отказавшись от совершенных и несовершенных консонансов пифагорейской гаммы. Сохранив октаву, А. Веркмайстер разделил ее

на 12 равных частей. Новый музыкальный строй позволил выполнять транспонирование мелодии. С введением этого строя в музыке восторжествовала темперация (от лат. соразмерность).

В чем же состояло математическое описание равномерно-темперированного строя? Анализ многих традиционных примеров [2], например, народной музыки показал, что чаще всего в ней встречаются интервалы, выражаемые с помощью отношений частот: 2 (октава), 3/2 (квинта), 5/4 (терция), 4/3 (кварта), 5/3 (секста), 9/8 (секунда), 15/8 (септима). Эти и другие выводы показали, что музыкальная шкала должна быть разделена на 12 частей и в математическом построении 12-и мажорные и 12-и минорные тональности тождественны.

Следует отметить, что до сих пор делаютсяп опытки усовершенствования равномерно-темперированного строя, в основу которого положены частоты, выражающиеся приближенными значениями чисел. А приближенное значение иррационального числа всегда определяется с заданной степенью точности. Итак, рождение нового музыкального строя не могло произойти без изобретения логарифмов и развития алгебры иррациональных величин [1]. Без знания логарифмов провести расчеты равномерно-темперированного строя было бы нереально. Логарифмы стали своеобразной «алгеброй гармонии», на которой выросла темперация. История создания равномерной темперации еще раз свидетельствует о том, как тесно переплетаются судьбы математики и музыки.

Приступая к анализу математических методов, следует отметить, что большая часть используемых понятий по своей природе нечетки и размыты, а попытка загнать их в границы комбинаторики или двоичной логики приводит к недопустимым искажениям. Можно только представить какие музыкальные оттенки, обертоны при таком подходе теряются навсегда. Не вдаваясь в детали, можно сказать, что в настоящее время известны несколько принципиально разных подходов к решению задач музыкального анализа. Во-первых, можно использовать классические методы анализа (например, корреляционные) – если данные взаимозависимы, а их объем относительно невелик. Во-вторых, можно построить экспертную систему. В-третьих, можно воспользоваться методами нейронных сетей. И наконец, как предлагается в настоящей работе, воспользоваться методами Fuzzy Logic (нечеткой логики). Для этого следует воспользоваться качественными характеристиками типа: «большинство (чуть больше)», «надежный (устойчивый)», «немного (чуть меньше)» и

т.п. Судьба нечеткой логики (Fuzzy Logic) как нового научного направления, во многом сходна с ее содержанием – необычна, сложна и парадоксальна. Нечеткую логику считают «третьей волной интеллектуального программирования». В ее основе лежит теория нечетких множеств, изложенная в серии работ Лотфи Заде. В этих работах рассматриваются элементы множеств, для которых функция принадлежности представляет собой не жесткий порог (принадлежит/не принадлежит), а плавную сигмоиду (часто упрощаемую ломаной линией), «пробегающую» все значения от нуля до единицы. Некоторые ученые полагают, что само название “fuzzy” («нечеткий», «размытый») применительно к теории Заде является не совсем адекватным и предлагают заменить его на более точное: «непрерывная логика».

Замечательные успехи, достигнутые с помощью методов нечеткой логики, позволили раскрыть многие тайны природы и создать все более и более совершенные алгоритмы, модели. Основной тезис создателей теории нечетких множеств заключается в том, что, по сути, обычные количественные методы анализа систем непригодны для гуманистических систем и, вообще, любых систем, сравнимых по сложности с гуманистическими. В основе этого тезиса лежит то, что можно было бы назвать принципом несовместимости. Суть этого принципа такова: чем сложнее система, тем труднее дать точные и, в то же время, имеющие практическое значение суждения о ее поведении. Для систем, сложность которых превосходит некоторый пороговый уровень, точность и практический смысл становятся почти исключающими друг друга характеристиками [4]. Альтернативный подход, предложенный Л. Заде, опирается на предположение о том, что в основе мышления человека «лежит не традиционная двузначная или даже многозначная логика, а логика с нечеткой истинностью, нечеткими связями и нечеткими правилами вывода». Этот подход дает приближенные, но весьма эффективные способы описания сложных систем, не поддающихся точному количественному анализу [5]. Самые простые правила которые можно будет использовать это – элементарные логические отношения (больше чем, и или, и т.д.), в то время как более сложные рассуждения о том, благоприятна ли ситуация или нет, будут строиться на основе более сложных отношений (если, то, иначе). Выбор интеллектуальных моделей с заданной степенью точности, оперируя принципами поведения системы, описываются Fuzzy-методами. Для описания систем можно воспользоваться и принципами обобщенной модели нечеткого вывода Мамдани и Такаги-Су-

гено, а также интереснейшими практическими работами Коско, который изобрел так называемые Fuzzy Cognitive Maps – нечеткие когнитивные модели, на которых базируется большинство современных систем динамического моделирования в финансах, политике, бизнесе, психологии и т.п. Им же была исследована взаимосвязь нечеткой логики и теории нейронных сетей.

Для любого анализа необходимо создавать базу данных, в этой связи хотелось бы отметить работы М. Земанковой, усилиями которой были заложены основы теории нечетких СУБД (систем управления базами данных), способных оперировать неточными данными, обрабатывать нечетко заданные запросы, а также использовать качественные параметры – наряду с количественными. Почти в то же время была разработана нечеткая алгебра – необычная наука, позволяющая использовать при вычислениях как точные, так и приблизительные значения переменных, что также является прекрасным инструментом при анализе нечетких, по своей природе, и размытых понятий. Названные способы вычислений, анализа и логического вывода получили широкое распространение. В области музыковедения обращение к теории нечетких множеств и ее инструментарию было впервые предпринято в 1998 году [6]. Апелляция именно к этому аналитическому аппарату разносторонне мотивирована. Например, дирижер, концертмейстер или аранжировщик заранее, до репетиции, прорабатывает интеллектуальную схему (с помощью теории Fuzzy Logic), исполнения, музыкального сопровождения или совместного с солистом создания художественного образа.

Следует отметить, что главная музыкальная мысль, заложенная в произведении – это мелодия, являющаяся основой музыки. Важнейший элемент музыки – ритм, а также характерная особенность – это чередование тяжелых звуков с более легкими. Ритм, мелодия, метр, гармония, тембр – в совокупности составляют язык музыки.

Поскольку, помимо использования нотного материала, а также для создания «впечатляющей звуковой картины-образа» возможны и желательны музыкальные импровизации, то ориентация на позицию профессионала, принимающего приблизительное решение по заранее проработанной им интеллектуальной схеме, основанной на теории Fuzzy Logic, в описанных условиях позволяет ввести нечеткое множество.

Задачи осмысления звукового образа и творческих возможностей, выявление путей реализации пытаются решить ученые в своих исследова-

тельских работах. С этой целью ниже приводятся основные понятия, используемые Л. Заде при описании своего подхода к анализу сложных систем и процессов принятия решений – нечеткие переменные, нечеткие высказывания и нечеткие алгоритмы. Следует отметить, что звуковые события, принадлежащие музыкальным произведениям (речевым произведениям на языке музыки), можно анализировать по аналогии с лексикой естественных языков. Каждое звуковое событие x может считаться сжатым «описанием» нечеткого подмножества M(x) от полного множества области рассуждений U, где M(x) есть значение x [6]. Так, например, x есть комплекс звуков, M(x) – видовая принадлежность этого комплекса (мажорный секстаккорд, малый минорный терцквартаккорд и т.д.). U – аккордика того или иного гармонического стиля. В обозначенном смысле язык музыки можно рассматривать как систему, в которой нечетким подмножествам множества U приписываются элементарные и составные символы (аккорды, гармонические обороты, тональные проследования и т.д.). Тогда с этой точки зрения можно представить звуковой комплекс, содержащий малую терцию и малую сексту от баса h. При этом значение интервального состава есть нечеткое подмножество M – мажорный секстаккорд, а значение той же интервальной конструкции в конкретной высотной позиции – нечеткое подмножество M – аккорд с основным тоном g. Таким образом, значение звукового комплекса h–d–g является пересечением M – мажорный секстаккорд и M – аккорд с основным тоном g. При этом в качестве нечеткой переменной рассматривается разновидность аккорда, тогда этой переменной (мажорный секстаккорд, малый уменьшенный квинтсекстаккорд и т.д.) могут быть символы нечетких подмножеств полного множества всех аккордов с основным тоном g. Вместе с тем переменной можно считать и сам основной тон, тогда ее значения (a, b, c и т.д.) следует интерпретировать как символы нечетких подмножеств полного множества всех тонов музыкальной системы. В этом смысле основной тон становится нечеткой переменной, т.е. переменной, значениями которой являются символы нечетких множеств. Важно отметить, что значение переменной «основной тон», выраженное музыковедческим понятием, гораздо менее точно, чем число, обозначающее частоту колебаний данного тона. В описанном случае рассматривается меньшая четкость теоретического понятия «тон» по сравнению с его акустическими параметрами. Нечеткость данного понятия неизмеримо возрастет, если принимать во внимание зонную природу музыкального слуха, в соответствии с

которой каждый тон представлен не единственной частотой колебаний, а некоторой частотной зоной.

Такой подход дает возможность количественного представления любых звуковых явлений, в том числе не имеющих в настоящее время апробированных средств измерения. Именно по этой причине многие специалисты теории музыки в свое время отказались от рассмотрения тембра, артикуляции, акцентов и других событий фонического уровня [7]. В количественных подходах к анализу систем простые отношения между двумя числовыми переменными обычно описывают с помощью таблицы, которую «словесно можно представить в виде набора высказываний, например, если x равен 5, то y равен 10; если x равен 6, то y равен 14 и т.д.» [6]. Такой же способ описания применяет и Л. Заде, только переменные ***x*** и ***y*** являются нечеткими [4]. Зависимость между ними описывается с помощью высказываний вида «из A следует B», где **A** и **B** – символы нечетких множеств, представляющие собой значения переменных ***x*** и ***y***. Можно привести пример нечетких высказываний, относящихся к рассматриваемой области музыкознания. Допустим, что [6] A – нечеткое множество, объединяющее значения переменной x (x1, x2 и т.д.), представляющей собой звуковысотную конструкцию, а B – нечеткое множество, объединяющее значения переменной y (y1, y2 и т.д.), представляющей собой оценку звуковысотной конструкции x слушателем.

Поскольку сложные зависимости ***y*** от ***x*** требуют для своего описания нечетких алгоритмов, то тогда нечеткий алгоритм будет представлять собой упорядоченную последовательность инструкций, некоторые из которых могут содержать символы нечетких множеств, например:

– если ***y*** велико, то немного уменьшить ***x***;

– если ***y*** не очень велико и не очень мало, то очень ненамного увеличить ***x***;

– если ***y*** мало, то стоп; если нет, то увеличить ***x*** на 2.

Использование аналитического аппарата теории нечетких множеств делает излишней разработку отдельных измерительных методик для каждого вида звуковых явлений. Названный аппарат позволяет оценивать указанные явления не с точки зрения количественной выраженности некоторых специфических качеств, а с точки зрения принадлежности этих качеств потоку художественной информации. Нечеткие высказывания, описывающие зависимость ***y*** от ***x***, могли бы выглядеть следующим образом:

– если ***x*** – минорное трезвучие с основным тоном h, то для x1 минорный

секстаккорд с основным тоном hy – событие с минимальной новизной (такое значение y объясняется тем, что x_1 является структурным вариантом x);

– если x_1 – минорный секстаккорд с основным тоном h, то для x_2 минорный секстаккорд с основным тоном c, y_1 – событие с большей степенью новизны (данное значение y_1 связано с тем, что x_2 является высотно-позиционным вариантом x_1);

– если x_2 – минорный секстаккорд с основным тоном c, то для x_3 – малый, уменьшенный терцквартаккорд с основным тоном e. При этомy_1 – уже новое событие (высокая степень новизны этого звукового события обусловлена тем, что x_3 является другим аккордом по отношению к x_2) и т.д. Здесь использованы только два варианта оценки звуковысотных конструкций, фиксирующих наличие или отсутствие события в гармоническом ряду.

Открывающаяся возможность ранжирования звуковых событий по степени их новизны позволит в случае ее реализации получить более детальное представление о событийной структуре музыкального произведения. С учетом актуальности исследования проблемы реализации творческих представлений [7] дирижера, интерпретатора-концертмейстера и солиста данная статья представляет собой попытку раскрыть объективные механизмы анализа музыкального произведения, гармонии, исполнительства, «технологию» творчества, практической реализации художественно-эстетических замыслов.

ЛИТЕРАТУРА

1. *Бонфельд М.Ш.* Анализ музыкальных произведений. Структуры тональной музыки в двух частях. М: Гуманитарный издательский центр «Владос», 2003.

2. *Масалович А.И.* Этот нечеткий, нечеткий, нечеткий мир М: PCWeek/REN. 16, 1995.

3. *Заде Л.А*. Основы нового подхода к анализу сложных систем и процессов принятия решений //Математика сегодня: сб. ст.; пер. с англ., Сост. А.В. Шилейко. М.,1974.

4. *Зубарева Н.Б.* О применении точных методов в анализе музыкальных произведений. Гуманитаризация образования и внеучебная работа в вузе, техникуме, образовательной школе: материалы IV межвуз. науч.-практ. конф. Пермь. 1998.

5. *Зубарева Н.Б.* О применении теории нечетких множеств в музыковедческом исследовании (на примере анализа фортепианного цикла С. Прокофьева «Детская музыка»). История и методология науки: межвуз. сб. науч. Трудов. Гл. ред. В. В. Маланин. Пермь. 2000. Вып. 7.

6. *Kulichkin P.A., Zubareva N.B.* Onthe formation role of the “events dynamics” in the musical composition for voice and piano. Proceedings of the International Congress on Aesthetics, Creativity and Psychology of the Arts. Perm 2005 Edited by E. Malianov, C. Martindale, E. Berezina, L. Dorfman, D. Leontiev, V. Petrov, P. Locher. Зубарева Н.Б., Калашникова И.С. «Событийная динамика» вокального произведения (опыт сравнительного анализа музыкального и поэтического рядов в романсах М. Глинки). М.: Вестник ПГИИК. 2005.

7. http://www.ug.ru/old/97.24/t8_1.htm

STUDY ON APPLICATION OF MATHEMATICAL METHODS FOR MUSIC WORKS’ ANALYSIS

L. Aznauryan

e-mail: lusine.aznauryan8@gmail.com

SUMMARY

This article is devoted to the research of mathematical methods detecting the objective mechanisms for description and analysis of music works, harmony, instrumental performance, art “technology” and implementation of artistic and aesthetic designs. The article a historical overview of mathematical methods discovered by Pythagoras and Archytas way back in 4th century before Christ, as well as by Johannes Kepler and Andreas Werckmeister, a German organist, in the Middle Ages. As a mathematical tool for sound event analysis the fuzzy-set theory methods are proposed to provide the basis for developing fuzzy interference algorithms of Fourier analysis.

Keywords: sounds and sound events, the ratio of pitch, harmonic intervals, tempered, fuzzy logic.

ԵՐԱԺՇՏԱԿԱՆ ՍՏԵՂԾԱԳՈՐԾՈՒԹՅՈՒՆՆԵՐԻ ՎԵՐԼՈՒԾՈՒԹՅԱՆ ՀԱՄԱՐ ՄԱԹԵՄԱՏԻԿԱԿԱՆ ՄԵԹՈԴՆԵՐԻ ԿԻՐԱՌՄԱՆ ՀՆԱՐԱՎՈՐՈՒԹՅԱՆ ՈՒՍՈՒՄՆԱՍԻՐՈՒՄ

Լ.Ա. Ազնաուրյան

Էլ.հասցե: lusine.aznauryan8@gmail.com

ԱՄՓՈՓՈՒՄ

Հոդվածը նվիրված է երաժշտական ստեղծագործությունների, հարմոնիայի, կատարումների, ստեղծագործման «տեխնոլոգիայի» և գեղարվեստա-գեղագիտական մտահղացումների կիրառական իրագործման նկարագրման և վերլուծության օբյեկտիվ մեխանիզմների բացահայտման մաթեմատիկական մեթոդների ուսումնասիրմանը: Կատարված է մ.թ.ա. IV դարում Պյութագորեսի և Արխիտեսի կողմից, ինչպես նաև միջնադարում Յ. Կեպլերի և գերմանացի երգեհոն Ա. Վերկմայստերի

կողմից բացահայտված մաթեմատիկական մեթոդների պատմական վերլուծությունը: Որպես ձայնի և ձայնային իրադարձությունների վերլուծության մաթե- մատիկական եղանակներ առաջարկվում են ոչ հստակ բազմությունների տեսության մեթոդները, որոնք հիմք կծառայեն ոչ հստակ եզրակացման մեթոդներով հարմոնիկ վերլուծության ալգորիթմների մշակման համար:

Հիմնաբառեր՝ Հնչյուններ եւ ձայնային իրադարձություններ, հնչյունների հարաբերակցություն, հարմոնիկ ընդմիջումներ, համաչափություն, Fuzzy Logic.

БИОЛОГИЯ

УДК 577.1 Поступила 18.11.2015г.

JNK/AP-1 СИГНАЛЬНЫЙ ПУТЬ ПРИ ИШЕМИЧЕСКОМ ИНСУЛЬТЕ

К. Тадевосян

Российско-Армянский (Славянский) университет,
Институт молекулярной биологии НАН РА
tadevosyankar@gmail.com

АННОТАЦИЯ

Ишемический инсульт (ИИ) является комплексным заболеванием, этиологическими факторами которого являются как генетические, так и факторы внешней среды. В патогенез ИИ вовлечено множество сигнальных путей, регулирующих процесс развития инсульта и период восстановления. Одним из таких сигнальных каскадов является JNK/AP-1. В зависимости от течения и тяжести ИИ JNK/AP-1 система, с одной стороны, участвует в патобиохимическом каскаде, с другой – активно вовлечена в восстановительные процессы после ИИ. Таким образом, звенья JNK/AP-1 сигнального каскада являются потенциальными мишенями для мониторинга и терапии при ИИ.

Ключевые слова*:* AP-1, JNK, ишемический инсульт.

Введение

Ишемический инсульт (ИИ) является наиболее распространенным острым неврологическим расстройством, составляющим 80% всех инсультов и 70% всех цереброваскулярных патологий (ЦСП). ИИ каждый год поражает 5–6,5 млн. человек и уносит 4,6 млн. жизней. 25% перенесших ИИ являются людьми активного возраста и только треть из них достигает полной социальной и профессиональной реинтеграции в общество, остальные либо умирают, либо остаются инвалидами до конца своих дней [1].

ИИ характеризуется внезапным нарушением кровообращения в области мозга в связи с сужением просвета ключевых артерий, либо их полной заку-

поркой, вследствие чего нарушается доставка кислорода в мозг с последующей потерей его неврологической активности. Главными причинами ИИ являются интракраниальный тромбоз или экстракраниальный эмболизм. Нарушение циркуляции крови в определенной области мозга приводит к запуску ишемического каскада, который ведет к смерти нейронов и церебральному инфаркту [2].

ИИ рассматривается как мультифакторное заболевание, так как причиной его могут служить как генетические факторы, так и факторы внешней среды [3, 4]. В патогенезе ИИ участвует множество сигнальных путей, которые регулируют процессы развития инсульта и восстановление организма после него. Имеется немало сведений о вовлеченности JNK/AP-1 сигнального пути в патогенез ИИ, звенья которого являются потенциальными мишенями для мониторингаи терапии ИИ.

JNK компонент

c-JunN-терминальная киназа (JNK) относится к семейству митоген-активирующихся протеинкиназ (MAPK) и является стресс-зависимой киназой, которая активируется в ответ на различные формы внешних раздражителей, включая ишемию [5–7]. Человек имеет 3 гена MAPK (MAPK 8, MAPK 9, MAPK 10), локализованных на хромосомах 10, 11, 4, соответственно, и каждый MAPK ген в ходе альтернативного сплайсинга образует белки со средней массой 45–46 кДа. JNK1 и JNK2 экспрессируются практически во всех органах и тканях, а JNK3 – в центральной нервной системе и сердечной мышце [8]. Каждый из ферментов имеет различную субстратную специфичность, вследствие чего выполняет определенную функцию в разных клеточных процессах. JNK1 участвует в физиологических процессах эмбрионального развития: в развитии нейрональных сетей, миграции нейронов, архитектуре дендридов и т.д. В свою очередь, JNK2 и JNK3 активируются в ответ на стрессовые стимулы. Основным субстратом JNK является c-Jun, который фосфолирируется по N-терминальному участку (серин 63 и 73). Однако многие исследования показали, что JNK также фосфолирует и другие транскрипционные факторы, такие, как c-Fos, ATF2 и т.д. Образующиеся активные формы транскрипционных факторов вместе с c-Jun являются компонентами димера активирующего белка-1 (AP-1), который регулирует экспрессию многих стресс-зависимых генов [9].

Показано, что все JNK играют важную роль как в процессе нейронального апоптоза, так и в воспалительных процессах, которые имеют место при ИИ [10, 11]. В частности, JNK регулируют транскрипцию генов, участвующих в отмеченных процессах, а также непосредственно связываются с этими генами. Например, JNK могут фосфолирировать *Bcl-2* и *Bcl-xL*, уменьшая их антиапоптозную активность. С другой стороны, фосфолирируя *Bim* и *Bmf*, они усиливают их проапоптотическую активность.

JNK является одной из главных терапевтических мишеней в лечении и диагностике ИИ. На животных моделях ИИ было продемонстрировано, что при использовании ингибиторов JNK уменьшается размер инфарктной зоны [6, 12].

AP-1 компонент

AP-1 – транскрипционный фактор состоит из гомо- и гетеродимеров, белков семейства Jun, Fos и ATF. Члены семейства AP-1 имеют структуру «лейциновой застежки-молнии» (BasicRegionLeucineZipper, bZIP), которая является эволюционно консервативной и характерной для многих транскрипционных факторов. С-терминальный домен «лейциновой застежки» ответственен за димеризацию, необходимую для осуществления регуляторных функций, а также за специфичность и стабильность димеров, а N-терминальный домен – за связывание транскрипционного фактора с ДНК. Димеры AP-1 вовлечены в регуляцию транскрипции многих генов, участвующих в процессе деления клеток, апоптозе и ответе на стрессовые сигналы, посредством связывания с консенсусными ДНК-регуляторными элементами. c-Jun гомодимеры и c-Jun/c-Fos гетеродимеры связываются с семинуклеотидной консенсусной последовательностью TGACTCA, также известную как TRE (TPAR esponsive Element), тогда как ATF 2 гомодимеры и c-Jun/-ATF 2 гетеродимеры связываются с восьминуклеотидной последовательностью TGACGTCA (CRE, c AMP Responsive Element) [13].

Роль JNK/AP-1 сигнального пути в развитии ишемического инсульта

Как уже было отмечено выше, в ходе ишемии происходит активация JNK/AP-1 сигнального каскада. В частности, по мере поступления сигнала происходит активация JNK посредством его фосфолирирования. Здесь особую роль играет JNK3, так как именно он ответственен за индукцию апоптотических процессов при стрессовых условиях. На животных моделях было показа-

но, что у мышей с мутантным геном JNK3 наблюдалась минимальная индукция рецептора «смерти» Fas. Это позволяет предположить, что JNK3 регулирует экспрессию главных участников апоптотического процесса [14]. В то же время, по мере активации JNK происходит фосфолириование c-Jun, который способен затем образовывать гомо- и гетеродимеры, тем самым комплектует AP-1. Было показано, что JNK-зависимая трансактивация c-Jun способствует нейрональному апоптозу в ходе ИИ [15]. Образование того или иного димера зависит от типа активирующего сигнала и времени его поступления. Как известно, в области ишемической полутени – ишемической пенумбре – происходят активные процессы нейроапоптоза, которые также могут быть обусловлены активацией JNK/AP-1 сигнального каскада, в частности, образованием c-Jun/ATF2 гетеродимера посредством активации JNK. Об этом также свидетельствуют результаты исследований, демонстрирующие, что использование малых РНК, образующих шпильки, которые ингибируют сборку димера c-Jun/-ATF2, предотвращает активацию нейроапоптоза [14, 16]. С другой стороны, увеличение экспрессии ATF2 подавляет образование c-Jun/c-Fos комплекса, который также образуется под воздействием JNK и участвует в клеточном выживании, что ведет к последующему нейрональному апоптозу. В свою очередь, сам c-Fos, помимо ингибирования образования комплекса c-Jun/-ATF2, также предотвращает взаимодействие этого гетеродимера с соответствующим сайтом на ДНК, тем самым, не допуская экпсрессию регулируемых им генов [17–19]. Данное событие можно объяснить тем, что в зависимости от тяжести ИИ отмеченный сигнальный каскад на начальных этапах активно вовлечен в процесс гибели нейронов, а при восстановительных процессах он участвует в их выживании.

Таким образом, в зависимости от тяжести ИИ JNK/AP-1 сигнальный каскад может участвовать как в процессе развития ИИ, так и в восстановительных процессах после него. Данное предположение является очень важным в аспекте мониторинга и терапии ИИ. В настоящее время проводятся многочисленные исследования, демонстрирующие положительные результаты относительно протекторной роли ингибиторов звеньев JNK/AP-1 сигнального каскада при ИИ [9].

ЛИТЕРАТУРА

1. *Biller J., Love B.B.,* Vascular diseases of the nervous system. A. Ischemic cerebrovascular disease. Neurology in clinical practice. The neurological disorders.2000; 2: 1125–1166.
2. *Harrigan M.R., Deveikis J.P.,* Handbook of cerebrovascular disease and neurointerventional technique. Humana Press, USA, 2009.
3. *Ikram M.A., Seshadri S., Bis J.C.,* et al. Genomewide association studies of stroke New Engl J Med. 2009; 360: 1718– 1728.
4. *Hugh S. Markus.,* Stroke genetics. Human Molecular Genetics. 2011; 20: 124– 131.
5. *Unal-Cevik I., Kılınc M., Can A., Gursoy-Ozdemir Y., Dalkara T.,* Diverse Roles of JNK and MKK Pathways in the Brain. Stroke. 2004; 35: 2189–2194.
6. *Kuan C., Burke R.,* Targeting the JNK signaling pathway for stroke and Parkinson's diseases therapy. Curr Drug Targets CNS Neurol Disord. 2005; 4: 63–67.
7. *Manning A., Davis* R., Target JNK for therapeutic benefit: from Junk to gold? Nat Rev Drug Discov. 2003; 2: 554–565.
8. *Tokiwa Y., Hiroshi K., Hiroshi N.,* Diverse Roles of JNK and MKK Pathways in the Brain. J. of Signal Transduction. 2012; 2012: 9.
9. *Jie C., Ming Z., Yong-qing Z., Zhi-heng* X., JNK pathway: diseases and therapeutic potential1. J.Acta Pharmacol. 2007; 28 (5): 601–608.
10. *Danny N. Dhanasekaran D., Reddy P.,* JNK Signaling in Apoptosis. Oncogene. 2008, 27, 6245–6251.
11. *Ip T., Davis* R. Signal transduction by the c-Jun N-terminal kinase (JNK) — from inflammation to development. Current Opinion in Cell Biology. 1998, 2, 205–219.
12. *Murata Y., Fujiwara N., HaeSeo J.,Yan F., Liu X.,* et al. Delayed Inhibition of c-Jun N-Terminal Kinase Worsens Outcomes after Focal Cerebral Ischemia. The Journal of Neuroscience. 2012,32, 8112– 8115.
13. *Hess J., Angel P., Schorpp-Kistner M.,* AP-1 subunits: quarrel and harmony among siblings // J Cell Sci. 2004. T. 117, № Pt 25. 5965–73.
14. *Kuan C., Whitmarsh A., Yang D., Liao G., Schloemer A., Dong C., Bao J., Banasiak K., Haddad G., Flavell R., Davis R., Rakic P.,* A critical role of neural-specific JNK3 for ischemic apoptosis. Proc Nat Acad Sci USA.2003, 100, 15184–15189.
15. *Raivich G.,Behrens A.,* Role of the AP-1 transcription factor c-Jun in developing, adult and injured brain. Prog Neurobiol. 2006, 78(6), 347–363.
16. *Ramos-CabrerP.,Campos F., SobrinoT.,Castillo J.,* Targeting the Ischemic Penumbra. Stroke. 2011, 42[suppl 1]:S7-S11.
17. *Dam van H., Castellazzi M.* ,Distinct roles of Jun:Fos and Jun:ATFdimers in oncogenesis. Oncogene. 2001, 20, 2453–2464.

18. *Zhongmin Y., Shoufang G., Jingyan L., Zhihao Z., Bin S.,* Opposing Roles for ATF2 and c-Fos in c-Jun-Mediated Neuronal Apoptosis. Molecular and Cellular Biology. 2009, 29, 2431–2442.

19. *Zhang, J.,Zhang D., Mc. Quade J., Behbehani M., Tsien J., and Xu M.,* c-fos regulates neuronal excitability and survival. Nat Genet. 2002, 30, 416–420.

JNK/AP-1 SIGNALING PATHWAY IN ISCHEMIC STROKE

K. Tadevosyan

SUMMARY

Ischemic stroke (IS) is a complex disorder caused by interplay of both environmental and genetic factors. There are a lot of signaling pathways involved in the pathogenesis of IS, which regulate stroke development and its repair. Growing evidence implicates the JNK/AP-1signaling pathway in the pathogenesis of IS. Depends on the severity of IS, JNK/AP-1 could participates as in pathobiochemical cascade as in the recovery processes after IS. So, members of the JNK/AP-1 signaling pathway are potential treatment and diagnosis targets of IS.
Keywords: AP-1, JNK, ischemic stroke.

JNK/AP-1 ԱԶԴԱԿԱՅԻՆ ՈՂՈՒ ԴԵՐԸ ԻՇԵՄԻԿ ԿԱԹՎԱԾՈՒՄ

Կ. Թադևոսյան

ԱՄՓՈՓՈՒՄ

Իշեմիկ կաթվածը (ԻԿ) հանդիսանում է համալիր հիվանդություն, որի առաջացման խթանիչներից կարող են լինել ինչպես ժառանգական, այնպես էլ արտաքին միջավայրի գործոնները: ԻԿ-ի պաթոգենեզում ներգրավված են բազմաթիվ ազդակային ուղիներ, որոնք կարգավորում են ԻԿ-ի զարգացման ընթացքը և վերականգնողական ժամանակահատածը: Այդպիսի ուղիներից մեկն է JNK/AP-1-ը: Կախված ԻԿ-ի ծանրությունից և ընթացքից, JNK/AP-1 համակարգը մի կողմից դեր է խաղում պաթոկենսաքիմիական կասկադում, մյուս կողմից էլ ակտիվորեն ընդգրկված է ԻԿ-ից վերականգնողական փուլի մեջ: Այսպիսով, JNK/AP-1 ազդակային ուղու օղակները հանդիսանում են ԻԿ մոնիտորինգի և թերապիայի պոտենցիալ թիրախ:
Հիմնաբառեր՝ AP-1, JNK, իշեմիկ կաթված:

УДК 502(479):06 Поступила 20.11.2015г.

ФОРМИРОВАНИЕ СИНАНТРОПИЗМА В НЕКОТОРЫХ СЕВЕРНЫХ РАЙОНАХ АРМЕНИИ НА ПРИМЕРЕ ПРЕДСТАВИТЕЛЕЙ ОТРЯДА ХИЩНЫХ МЛЕКОПИТАЮЩИХ

Л.Г. Папян

Российско-Армянский (Славянский) университет
lyov.papyan@gmail.com

АННОТАЦИЯ

Процесс формирования синантропизма у представителей разных отрядов млекопитающих во временном разрезе протекал с заметными отклонениями в связи с образом жизни некоторых видов млекопитающих разных регионов Армении, о чем свидетельствуют наши данные, связанные с наблюдением за некоторыми из мелких видов хищных животных.

Ключевые слова: синантропизм, фотоловушка, хищники.

Введение

Различные природные и антропогенные факторы – такие, как хищническая вырубка леса, повышение уровня озера Севан, преследование и обстрел хищных видов зверей и птиц-пожирателей грызунов и насекомых привело к тому, что некоторые виды животных были вынуждены спасаться, стараясь жить рядом, или нередко под одной крышей с тем же человеком. Человеческие постройки, в особенности многоэтажные дома, в которых функционируют мусоропроводы, являются для солидного числа видов животных прекрасным и удобным местом для жилья, питания и размножения.

Представители отряда грызунов, всегда в той или иной степени, являлись синантропами. Они давно приспособились к сожительству с человеком, и в течение времени отлично адаптировались к новым условиям. Но в последнее время все чаще и чаще из различных регионов Армении поступают сообщения о том, что в пределах человеческих поселений, и даже городов, днем были замечены представители отряда хищных. А недавно в Ереване, в окрестностях Норкского массива, была замечена лисица (Vulpes vulpes). В наших исследованиях мы пытаемся установить причину таких явлений, определить

видовой состав живонтых-синантропов и изучить изменения в их поведении и суточной активности. В данной статье были исследованы поведенческие особенности представителей отряда хищных млекопитающих в некоторых северных районах Армении.

МАТЕРИАЛЫ И МЕТОДЫ

Есть предположение, что хищники приходят в человеческие поселения за пищей. Многие из них питаются отбросами человека и грызунами и другими мелкими млекопитающими, которые давно перешли в синантропный образ жизни. Объектом исследования являлись представители отряда лесных и полевых мышей (Apodemus) и хищных (Carnivora).

Установление фотоловушек

Чтобы убедиться в правдивости данного предположения, мы, начиная с марта 2015г., устанавливали фотоловушки в северных районах Армении. В ходе работы мы использовали фотоловушки 940 NM HD 2013 с сенсорными датчиками, реагирующие на движения животного. Фото и видео материалы получились высокого разрешения (720p).

Устройства были установлены на территории следующих регионов: Гехаркуник, Котайк, Лори, Тавуш. В каждом отдельном участке устройства работали круглосуточно. Некоторые из них работали в режиме ожидания и начинали фото- или видео-съемку только тогда, когда вспышка реагировала на движение какого-либо животного, а вторая группа устройств снимала непрерывно.

В качестве приманок были использованы мясо, хлеб и колбаса.

Результаты и обсуждение

На данный момент у нас имеются первые и весьма неплохие результаты. В двух регионах нам удалось зафиксировать появление каменной куницы (Martes foina) вблизи человеческих построек. Сначала в Агавнадзоре (Котайк) наши устройства зарегистрировали появление животного рядом со студенческим пансионатом ЕГПУ. На кадрах видно, как лесная мышь (Apodemus) приходит к приманке, а потом резко сбрасывает и убегает в лес. Через несколько секунд в кадре появляется куница и начинает преследовать свою жертву. Второй раз наши устройства зарегистрировали куницу на территории Севанского ботанического сада. На этот раз сам хищник приходил за приманками.

Из полученного нами материала мы сделали вывод, что, по сути, существуют 2 схемы появления хищников в поселках. В первом случае хищники

приходят в города и села за мелкими формами синантропов, которые для них являются прекрасной пищей. Во втором случае хищник сам питается нашими приманками, а это в лишний раз подтверждает тот факт, что хищники тоже приспособились к человеку.

Заключение

Полученные нами результаты доказывают, что все сообщения, котороые мы получаем из различных регионов нашей республики, правдивы. И результаты эти весьма тревожные. Если синантропность хищников на данный момент заключается только в том, что они преследуют своих жертв и оказываются в человеческих поселениях или же питаются отбросами человека, то что будет в ближайшем будущем, когда из-за неправильного и губительного использования природы человеком этим животным будет негде жить и они, спасаясь, окончательно перейдут в синантропный образ жизни или же окажутся на грани вымирания.

ЛИТЕРАТУРА

1. *Папян Л.Г., Арутюнян М.К., Гуланян В.Г., Асланян А.Г.* Увеличение числа синантропных видов позвоночных в бассейне озера Севан в связи с повышением его уровня // Материалы международной научной конференции «Биологическое разнообразие и проблемы охраны фауны Кавказа-2». 2014. СС. 286–287.

2. *Папян Л.Г., Гамбарян Г.Г.* Исследование поведения и активности мелких млекопитающих методом использования фотоловушек. Ломоносов-2015, XXII международная конференция студентов, аспирантов и молодых ученых: Сер: «Биология». 2015. С. 171.

3. *Саакян М.С.* Фауна грызунов северо-восточной Армении // Труды Арм. противочумной станции. 1964, Вып. 3. СС. 329–346.

4. *Сидорчук Н.В.* и др. Опыт использования фотоловушек при изучении поведенческой экологии барсука Meles meles. Териофауна России и сопредельных территорий: матер. VIII съезда Териологич. общ-ва. 2007. С. 455.

5. *Эрнандес-Бланко и др.* Опыт применения цифровых фотоловушек для идентификации Амурских тигров, оценки их активности и использования основных маршрутов перемещений животными // В кн.: Амурский тигр в Северо-Восточной Азии: проблемы сохранения в XXI веке. 2010. СС. 100–103.

6. *Сидорчук Н.В., Рожнов В.В.* Дистанционные методы изучения барсуков: некоторые особенности использования фотоловушек //Дистанционные методы изучения в зоологии: матер. научн. конф. 2011. С. 87.

ՀԱՅԱՍՏԱՆԻ ՈՐՈՇ ՀՅՈՒՍԻՍԱՅԻՆ ՄԱՐԶԵՐՈՒՄ ՍԻՆԱՆԹՐՈՊԻԶՄԻ ՁԵՎԱՎՈՐՈՒՄԸ ԳԻՇԱՏԻՉ ԿԱԹՆԱՍՈՒՆՆԵՐԻ ՕՐԻՆԱԿՈՎ

Լ.Հ. Պապյան

Հայ-Ռուսական (Սլավոնական) համալսարան
lyov.papyan@gmail.com

ԱՄՓՈՓՈՒՄ

Սինանթրոպիզմի ձևավորման գործընթացը կաթնասունների տարբեր ընտանիքների ներկայացուցիչների մոտ երկար ժամանակահատվածում ընթացել է նկատելի շեղումներով՝ կախված նրանց տեսակային առանձնահատկություններից ու վարած կենսակերպից: Սրա մասին են խոսում որոշ մանր կաթնասունների նկատմամբ մեր կողմից անցկացված ուսումնասիրությունների արդյունքում ստացած տվյալները:
Հիմնաբառեր՝ սինանթրոպիզմ, լուսանկարահանող սարք, գիշատիչ:

THE FORMATION OF SYNANTHROPISM WITH THE EXAMPLE OF SOME CARNIVORES IN SEVERAL NORTHERN REGIONS

L. Papyan

Russian-Armenian (Slavonic) University
lyov.papyan@gmail.com

SUMMARY

The formation of synanthropism of different mammal species has occurred during the time and has depended on their specifications and lifestyle. The results of our researches with the example of some small carnivores are the best evidence for it.
Keywords: synanthropism, trail camera, predators.

УДК 577.322 Поступила 20.10.2015г.

DATABASES FOR COMPLETE HUMAN MITOGENOMES: A REVIEW

H. Hovhannisyan

Russia-Armenian University, Institute of Molecular Biology NAS RA
e-mail: grant.hovhannisyan@gmail.com

SUMMARY

The progress of molecular genetics in the past two decades is characterized by the extensive improvement of DNA sequencing methods and rapidly accumulating data on genetic variation of different species. In particular, a big amount ofdata on mitochondrial DNA (mtDNA), which is invaluable for evolutionary, population genetics and forensic studies, have greatly improved our understanding of theanatomically modernhuman dispersal and allowed the reconstruction of matrilineal genetics histories of numerous human populations. For handling and analyzingthisimmensevolume of human mitogenomicinformation numerous specific databases were recentlydesigned. Here, wereview on the most commonly used resources and highlight advantages anddrawbacks that should considered while implementingnew databases for mitogenomic data management.

Keywords: human mitochondrial DNA, database, population genetics.

Introduction

Mitochondria are mammalian cellular organelles that have the function of oxidative phosphorilation and formation of ATP. Two distinct genetic systems encode mitochondrial proteins: mitochondrial and nuclear DNA. mtDNA is a small, 16 569 base pairs (b.p.), circle of double-stranded DNA which encodes 13 essential components of the respiratory electron transport chain, 2 ribosomal (12S and 16S) and 22 transfer RNA’s (Fig. 1).It is inherited maternally [1], thus does not recombine with paternal molecule [2,3], has a fast mutation rate [4] and presented by multiple copies in cell [5,6].

These unique featuresmade mtDNA a versatile tool for evolutionary and population genetics studies in the end of last century, when genome-wide or whole genome population studies were a daydream for researcher.

After the publication of the milestone Cambridge reference sequence by Anderson and colleagues[7] researchers began studying human mtDNAusing low resolution restriction fragment length polymorphism analysis with only a few restriction enzymes, utilizing them to estimate very simple phylogenetic trees. After the application of higher-resolution restriction analysis approaches on human mtDNA, first implemented at the Wilson's group(UC Berkeley, USA [8]), mitochondrial genome has becomepopular tool for population geneticists. Finally, in 2000th, the development of fully automated sequencing technologies allowed massive sequencing of whole mtDNA genomes, which has opened a new phase of mtDNA research. Since then numerous studies have been published where the authors have sequenced hundreds or even several thousand [9-11] humanmitogenomes in order to address medical issues [12], different questions of population genetics [13], phylogeography [14], demography [15], etc.

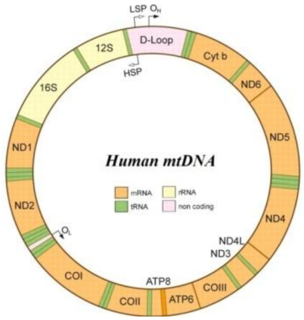


Figure 1. The structure of human mtDNA. Genes are indicated in orange, rRNA's – in yellow, tRNA's – in green, non-coding regions – in violet. LSP – light strand promoter, HSP – heavy strand promoter, O_L – origin of light strand, O_H – origin of heavy strand.

According to *Phylotree* database [16], which defines current nomenclature of global human mitochondrial DNA variation, to the date of February 17th2014 20666 complete human mitogenomes were sequenced and published in ca 300 papers and projects.Despite these amount of datadoes not exceed even 1/10 of entire human genome, there are numerous features ofhuman mitogenomes, such as mtDNAhaplogroups, their geographic distribution, ethnicity of individuals screened, etc., which make problematic a large-scale mtDNA data management and analysis. To integrate these data for further handling and effective analysis, in a recent decade several attempts were made do implement databases designed specifically for human mitochondrial DNA data. In this paper, we review the most utilized resources that weredesigned for human mitogenomic data management.

Overview of databases

HvrBase

One of the first databases for these specific kind of data was *HvrBase* [17] launched in 1998.It contained aligned sequences of the hypervariable regions I and II (HVRI and HVRII) with available information on the individuals (humans or apes) from whom the sequences were obtained. The authors implemented a 'search' function, allowing to retrieve the sequences matching a key-word defined byusers, enabling them to query the following types of information: name in publication or GenBank, author and publication, species, population,continent and origin. Additionally, sequences were searchable for certain motifs.

The collection of human data at the time of publication of the *HvrBase*comprised 5846 and 2302 HVRI and HVRII sequences,respectively. For 2061 samples, the both of HVR regions were sequenced. *HvrBase* was not updated since 2000, and, moreover, today, when researchers utilize complete mitochondrial genomes for population genetics studies, the data on HVR1 and HVR2 arenot widely used anymore.

In 2005,by adding more sequences from primates including humans, the database was updated to *HvrBase++*[18] – the improved and extended version of *HvrBase*. The *HvrBase++* database comprised not only the data on hyper-variable regions but also the mitochondrial genomes and nuclear sequences from several chromosomal loci. Human datawere presented by 20,037 sequences. Table 1 displays a human HVRI dataset gathered from 103 publications which encompasses sequences from 89 countries and 220 ethnic groups.

Table 1. Human HVRI datasets over six continents

Continent	Lineages	Human samples	Number of countries	Populations	Languages
Europe	2033	4358	17	25	31
Africa	1046	1680	25	47	47
North America	824	1581	7	34	9
South America	267	473	7	11	19
Asia	2867	4778	23	102	67
Australia/Oceania	224	473	10	12	28
World	7036	13343	89	220	194

The collection comprised 13 873 HVRI and 4940 HVRII sequences. Additionally, authors included 1376 complete mitochondrial genomes, 205 sequences from X-chromosomal loci and 202 sequences from autosomal chromosomes 1, 8, 11 and 16. In order to reduce the introduction of erroneous data into *HvrBase*++, the authors have developed a procedure that monitored GenBank for new versions of the current data in *HvrBase*++ and automatically updated the collection. For the stored sequences, supplementary information on the donors, such as geographic origin, population affiliation and language could be retrieved. As a new key feature, *HvrBase*++ provided an interactive graphical tool to easily access data from dynamically created geographical maps.Now the *HvrBase*++ is outdated and not available through the link (http://www.hvrbase.org/) provided by the authors of the database.

mtDB

The *mtDB* [19] database was created as a compendium for human mtDNA sequences.It was launched in 2005 and provided users with population genetic and medical data. *mtDB* had three main types of functional features. First, it allowed downloading of all mtDNA sequences either as individuals or population sets. All datawere grouped into 10 major geographic regions based on the population affiliation of the donors. Datasets from the same ethnic groups were available as batches of individual files. All sequences were cross-referenced to their original publications and most part of them was linked to theirGenBank accession numbers.By the 1 March 2007 there were 2697complete mitogenomes taken from 33 references (Table 2).

Table 2: Number of complete human mitogenomes obtained from corresponding geographic regions

Geographic region	Number of complete mitogenomes
Africa	287
Middle East	45
Europe	1192
North America	11
South America	14
Asia	922
South Asia	125
Australia	32
Melanesia/Micronesia/Polynesia	69
Total	**2697**

Secondly, *mtDB* had a function of finding polymorphic sites. Atthe moment of *mtDB* publication, 3311 polymorphic sites were identified and characterized in tabular form. This table comprised a separate line for each variable site with a count of how many sequences contain each particular nucleotide variant at that site, genetic location of the site, codon number and position, and details of amino acid changes. Clicking on the number of a particular variant users were able to obtain a list of the sequences, containing that particular mutation (insertions relative to CRS were discarded) and then to download the sequences.

Third, a'search' function for mitochondrial haplotypes was implemented in the *mtDB*. Haplotypes were searchable by entering the '*position'* and '*nucleotide'* for up to 10 loci. Again, these sequences could then be downloaded from the database.

mtDB was outdated since March 1, 2007 and the lack of convenient functionality of sequence retrieval and data parsing limited the usage of this database.

HmtDB

The database was created in 2005 [20] and relaunched in January 2012 [21].

HmtDB stores mitogenomic data annotated with population and variability information (Table 3).

Table 3: total amount of mitogenomic data in HmtDB

Geographic region	Individual type	Number of genomes	Complete genomes	Only coding region genomes
Africa	Normal	2323	2175	148
	Patient	71	71	0
America	Normal	1692	1587	105
	Patient	26	24	2
Asia	Normal	5715	5655	60
	Patient	1115	1115	0
Europe	Normal	7142	6648	494
	Patient	1706	1569	137
Oceania	Normal	1539	1524	15
	Patient	0	0	0
Undefined Continent	Normal	5128	5102	26
	Patient	524	521	3
All continents	Normal	23539	22691	848
	Patient	3442	3300	142

The annotations of sequences are curated manually, which provides higher accuracy of the data.Authors have designed a 'Classifier tools' that allow the database to predict the haplogroups, bases on *Phylotree*, for all mitogenomes stored in database or new sequences provided by users.

HmtDB provides with three main categories of usage: first, users can browse the database by multi-criterion 'query' system; second, analyze their own human mitochondrial sequences via the 'classify' tool (for complete genomes) or by downloading the 'fragment-classifier' tool (for partial sequences) and third, end-users can download sequence alignments with reference genomes as well as variability data.

Summarizing, we have reviewed three main databases devoted for human mitochondrial DNA storage and management – *HvrBase++*, *mtDB* and *HmtDB*.

HvrBase++ [4] and *mtDB* [5] databases were launched in ten years ago but have not been updated since 2007, while the number of new mtDNA partial and complete sequences has increased significantly since then. Besides of being outdated, these databases did not have appropriate functional characteristics for managing mtDNA data. For example, a lack of the data on human mtDNA haplogroup and sorting/grouping functions were restricting the usage of *HvrBase++* and *mtDB*. Additionally, today, when the Next-generation technologies allow massive sequencing of entire mitochondrial genomes on population scale, the data on HVRI and HVRII of mtDNA available in these databases, do not fulfill the requirements of modern population genetics.

On the other hand, the database *HmtDB* has numerous options for complex data searching –a powerful querying system, where user can search data according to the mutated position, haplogroup, geographic region, tissue, sex, etc; mtDNA-haplogroup assignment tool, convenient downloading function.However, it does notcontain sequences obtained after 2013, which also does not allow researchers to handle all the available mt DNA data.

Concluding, despite numerous databases for human mitogenomic data management were designed in a recent decade, one part of them is not curated and updated regularly, while the other do not provide convenient functionality of effective data parsing and further analysis.

REFERENCES

1. *Hutchison, C.A., Newbold, J.E., Potter, S. S.&Edgell, M.H.* (1974). Maternal inheritance of mammalian mitochondrial DNA. Nature, 251(5475): 536–8.
2. *Brown, W.M., George, M., &Wilson, A.C.* (1979). Rapid evolution of animal mitochondrial DNA. *Proceedings of the National Academy of Sciences, USA*, 76(4), 1967–1971.
3. *Michaels, G.S., Hauswirth, W.W., &Laipis, P.J. (*1982). Mitochondrial DNA copy number in bovine oocytes and somatic cells. *Developmental Biology*, 94(1), 246–251.
4. *Pikó, L., &Matsumoto, L.* (1976). Number of mitochondria and some properties of mitochondrial DNA in the mouse egg. Developmental Biology, 49(1), 1–10.
5. *Hagström, E., Freyer, C., Battersby, B.J., Stewart, J.B.,&Larsson, N.G*. (2014). No recombination of mt DNA after heteroplasmy for 50 generations in the mouse maternal germline. *Nucleic Acids Research*, 42(2), 1111–1116.
6. *Merriwether D.A., Clark A.G., Ballinger S.W., Schurr T.G., Soodyall H., Jenkins T., Sherry S.T.&Wallace D.C.* (1991). The structure of human mitochondrial DNA variation. *Journal of Molecular Evolution*, 33(6), 543–555.
7. *Anderson S., Bankier A.T., Barrell B.G., de Bruijn M.H., Coulson A.R., Drouin J., Eperon I.C., Nierlich D.P., Roe B.A., Sanger F., Schreier P.H., Smith A.J., Staden R., Young I.G*. (1981). Sequence and organization of the human mitochondrial genome. Nature, 290(5806):457–65.
8. *Cann R.L., Stoneking M., Wilson A.C.* (1987) Mitochondrial DNA and human evolution. *Nature* 325: 31–36.
9. *Derenko, M., Malyarchuk, B., Bahmanimehr, A., Denisova, G., Perkova, M., Farjadian, S., &Yepiskoposyan, L.* (2013). Complete mitochondrial DNA diversity in Iranians. PLoS*One*, 14;8(11):e 8067.
10. *Zheng, H.X., Yan, S., Qin, Z.D., &Jin, L*. (2012). MtDNA analysis of global populations support that major population expansions began before Neolithic Time. Scientific Reports, 2:745
11. *Behar D.M.*, van *Oven M., Rosset S., Metspalu M., Loogväli E. L., Silva N. M., Kivisild T., Torroni A., &Villems R.* (2012). A “Copernican” reassessment of the human mitochondrial DNA tree from its root. *American Journal of Human Genetics*, 90(4), 675– 684.
12. *Brandon, M., Baldi, P.A., &Wallace, D.C.* (2006). Mitochondrial mutations in cancer. *Oncogene*, *25*(34), 4647–4662.
13. *Duggan, A.T., Evans, B., Friedlaender, F.R., Friedlaender, J.S., Koki, G., Merriwether, D. A., Kayser M.*&Stoneking, M. (2014). Maternal history of Oceania from complete mtDNA genomes: contrasting ancient diversity with recent homogenization due to the Austronesian expansion. *The American Journal of Human Genetics*, 94(5), 721–733.
14. *Derenko, M., Malyarchuk, B., Denisova, G., Perkova, M., Litvinov, A., Grzybowski, T., Dambueva I., Skonieczna K., Rogalla U., Tsybovsky I. &Zakharov, I.* (2014). Western Eurasian ancestry in modern Siberians based on mitogenomic data. BMC Evolutionary Biology, 14(1), 217.

15. *O'Fallon, B.D., & Fehren-Schmitz, L.* (2011). Native Americans experienced a strong population bottleneck coincident with European contact. *Proceedings of the National Academy of Sciences*, USA, 108(51), 20444–20448.

16. *Van Oven, M., &Kayser*, M. (2009). Updated comprehensive phylogenetic tree of global human mitochondrial DNA variation. *Human Mutation*, 30(2), E386–E394.

17. *Burckhardt, F., von Haeseler, A.,&Meyer, S*. (1999). HvrBase: compilation of mtDNA control region sequences from primates. *Nucleic Acids Research*, *27*(1), 138–142.

18. *Kohl, J., Paulsen, I., Laubach, T., Radtke, A.,&von Haeseler, A.* (2006). HvrBase++: a phylogenetic database for primate species. *Nucleic acids research*, *34*(suppl 1), D700–D704.

19. *Ingman, M.*, &Gyllensten, U. (2006). Mt DB: Human Mitochondrial Genome Database, a resource for population genetics and medical sciences. *Nucleic Acids Research*, 34(suppl 1), D749-D751.

20. *Attimonelli, M., Accetturo, M., Santamaria, M., Lascaro, D., Scioscia, G., Pappadà, G., Russo L., Zanchetta L. &Tommaseo-Ponzetta*, M. (2005). HmtDB, a human mitochondrial genomic resource based on variability studies supporting population genetics and biomedical research. *BMC Bioinformatics*, 6(Suppl 4), S4.

21. *Rubino, F., Piredda, R., Calabrese, F.M., Simone, D., Lang, M., Calabrese, C., Petruzzella V., Tommaseo-Ponzetta M., Gasparre G.& Attimonelli*, M. (2012). Hmt DB, a genomic resource for mitochondrion-based human variability studies. Nucleic Acids Research, 40(D1), D1150–D1159.

ՄԱՐԴՈՒ ՄԻՏՈԳԵՆՈՄՆԵՐԻ ՏՎԱԼՆԵՐԻ ԲԱԶԱՆԵՐ՝ ԸՆԴՀԱՆՈՒՐ ԱԿՆԱՐԿ

Հ. Հովհաննիսյան

Հայ-Ռուսական համալսարան,
ՀՀ ԳԱԱ Մոլեկուլային կենսաբանության ինստիտուտ
e-mail: grant.hovhannisyan@gmail.com

ԱՄՓՈՓՈՒՄ

Վերջին երկու տասնամյակների ընթացքում մոլեկուլային գենետիկայի նվաճումները բնութագրվում են ԴՆԹ-ի սեկվենավորման մեթոդների էական կատարելագործմամբ և կենսաբանական զանազան տեսակների գենետիկական փոփոխականության տվյալների արագ կուտակմամբ: Մասնավորապես, միտոքոնդրիումային ԴՆԹ-ի (մտԴՆԹ) վերաբերյալ մեծաթիվ տվյալները, որը կարևոր նշանակություն ունի էվոլուցիոն, պոպուլյացիոն գենետիկական և դատաբժշկական հետազոտություններում, հնարավորություն են ընձեռել ընդլայնել մեր պատկերացումները ժամանակակից մարդու սփռման և բնակեցման հարցերի մասին, ինչպես նաև վերականգնել բազմա-

թիվ պոպուլյացիաների մայրագծային գենետիկական պատմությունը: Այդպիսի հսկայածավալ միտոգենոմային տեղեկությունների պահպանման և վերլուծության նպատակով վերջին տարիներին ստեղծվել են մի շարք տվյալների բազաներ: Սույն վերլուծական ակնարկում ամփոփ ներկայացված է առավել հաճախ օգտագործվող միտոգենոմային տվյալների բազաների նկարագրությունը՝ ընդգծելով դրանց առավելությունները և թերությունները, որոնք անհրաժեշտ է հաշվի առնել միտոգենոմային տվյալների կառավարման համանման գործիքներ ստեղծելիս:

Հիմնաբառեր՝ մարդու միտոքոնդրիումային ԴՆԹ, տվյալների-բազա, պոպուլյացիոն գենետիկա:

ОБЗОРБАЗ ДАННЫХ ДЛЯ МИТОГЕНОМОВ ЧЕЛОВЕКА

Г. Оганесян[1,2]

[1]*Российско-Армянский университет*
[2]*Институт молекулярной биологии НАН РА*
e-mail: grant.hovhannisyan@gmail.com

АННОТАЦИЯ

Успехи молекулярной генетики за последние два десятилетия характеризуются значительнымусовершенствованиемтехники секвенирования ДНК и быстрым накоплением информации о генетической вариабельностиразличных биологических видов. В частности, рост данных о митохондриальной ДНК (мтДНК), которая имеет важное значение для эволюционных, популяционно генетических и судебно-медицинских исследований, позволилирасширить наше понимание проблемы расселения современного человека и реконструировать матрилинейную генетическую историю многочисленных популяций человека. Для хранения и анализа большого объема информацииомитохондриальных геномах человека в последние годы были созданы многочисленные специализированные базы данных. В данной работепредставлен обзор наиболее частоиспользуемых баз митохондриальной ДНК человека с анализом ихпреимуществ и недостатков, которые необходимо учитывать при создании аналогичных инструментовдля менеджмента митогеномной информации.

Ключевые слова: митохондриальная ДНК человека, базы данных, популяционная генетика.

СВЕДЕНИЯ ОБ АВТОРАХ

С.А. Амбарцумян – академик НАН РА, иностранный член РАН

Л.А. Азнаурян – старший преподаватель кафедры системного программирования РАУ

М.В. Белубекян – к.ф.-м.н., профессор, главный научный сотрудник Института механики НАН РА

Г. Геворкян – аспирант третьего года обучения кафедры системного программирования РАУ

Д. Даноян – аспирант третьего года обучения кафедры дискретной математики и теоретической информатики ЕГУ

М.Г. Манукян – к.ф.-м.н., доцент

Л.Б.Овакимян – к.ф.-м.н., старший научный сотрудник Института радиофизики и электроники НАН РА

М. Овсепян – аспирант третьего года обучения кафедры системного программирования РАУ

Г.Г. Оганесян – аспирант второго года обучения кафедры биоинженерии и биоинформатики РАУ

Л.Г. Папян – аспирант первого года обучения Института зоологии НАН РА

Т. Сохакян – аспирант третьего года обучения кафедры системного программирования РАУ

К.М. Тадевосян – аспирант третьего года обучения кафедры медицинской биохимии и биотехнологии РАУ

К СВЕДЕНИЮ АВТОРОВ

Правила для авторов журнала «Вестник РАУ, Физико-математические и естественные науки».

Журнал печатает оригинальные статьи по различным направлениям физико-математических и естественных наук.

• К рассмотрению принимаются статьи на русском, армянском или английском языках.

• Статьи должны быть представлены в распечатанном виде и электронной форме.

• К материалам статьи прилагается Договор с издательством РАУ, подписанный одним (ответственным) автором (оформляется в одном экземпляре).

• Статья должна иметь направление от учреждения, в котором выполнена работа. Рукопись подписывается автором (соавторами) с указанием фамилии, имени, отчества, домашнего адреса, места работы, номеров телефонов и e-mail. Необходимо указать, с кем вести переговоры и переписку. Отклоненные статьи не возвращаются.

• В редакцию направляются два экземпляра статьи, набранные шрифтом 12 пунктов через 1.5 интервала на одной стороне листа. Рукописные вставки не допускаются. Все страницы должны быть пронумерованы.

Перед текстом статьи указываются:

– название статьи;

– инициалы и фамилии авторов (для иностранных авторов на языке оригинала или на английском языке);

– название учреждения (без сокращений и аббревиатур), которое направляет статью, его адрес (город, страна);

– e-mail авторов.

Далее помещается аннотация на языке оригинала объемом не более 0.5 машинописной страницы, которая не должна дублировать вводный или заключительный разделы. Аннотация не должна содержать литературных ссылок и аббревиатур. В конце аннотации указываются ключевые слова (keywords). В конце статьи помещаются аннотации на двух из оставшихся языках

• Изложение материала должно быть ясным и кратким, без формул и выкладок промежуточного характера и громоздких математических выражений.

• Рисунки, помещенные в тексте статьи, должны быть достаточно яркими и контрастными, чтобы сохранилось их качество при тиражировании журнала. Подрисуночный текст обязателен и должен быть набран курсивом.

• Формулы следует набирать курсивом, крупно, свободно и четко (набор математических формул рекомендуется выполнить при помощи системы Mathtype). Нумерация формул должна быть сквозной по всей статье (не по разделам).

– Жирным шрифтом набираются только векторные величины (стрелка сверху не нужна).

– Химические формулы, символы, сокращения, единицы измерения набираются прямым шрифтом.

• Таблицы должны быть включены в общую нумерацию текста. Обязательно наличие заголовков и единиц измерения величин. Все столбцы таблицы должны быть озаглавлены.

• Список литературы должен быть набран на языке цитированной литературы и оформлен следующим образом:

– для книг – инициалы и фамилии *всех* авторов, название книги, издательство, место издания, год издания в круглых скобках, том;

– для периодических изданий – инициалы и фамилии всех авторов, название журнала, том, – номера первой и последней страниц статьи, год издания в круглых скобках.

С О Д Е Р Ж А Н И Е

Адрес Редакции научных изданий Российско-Армянского
(Славянского) университета:
0051, г. Ереван, ул. Овсепа Эмина, 123
тел./факс: (+374 10) 27-70-52, (внутр. 42-02)
e-mail: marvolskraya@gmail.com

Заказ № 1
Подписано к печати 19.12.2015г.
Формат 70х100$^1/_{16}$. Бумага офсетная №1.
Объем 6.7 усл. п.л. Тираж 100 экз.